Jani Arponen

Teollisen internetin tiedonsiirto pilvipalveluun ja historiatietokannan käyttö pilvivarastona

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Automaatiotekniikka

Insinöörityö

8.5.2018

Metropolia

Tekijä Otsikko Sivumäärä Aika	Jani Arponen Teollisen internetin tiedonsiirto pilvipalveluun ja historiatietokannan käyttö pilvivarastona 47 sivua 8.5.2018
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	Automaatiotekniikka
Ammatillinen pääaine	
Ohjaajat	IoT Solutions Manager Jukka Pirinen Lehtori Timo Kasurinen

Tämä insinöörityö tehtiin Novotek Oy:lle. Novotek toimittaa teollisuuden IT- ja automaatioratkaisuja ja -ohjelmistoja. Työn tarkoituksena oli tutustua yrityksen tiedonsiirtoratkaisuihin kentältä pilvipalveluun sekä ratkaista ongelma GE Historian -historiatietokannan käytössä Azure-pilvipalvelussa.

Työn alussa käydään läpi teollisen internetin tiedonsiirtoa, tietoturvaa sekä pilvialustoja teoriapohjalta sekä esitellään kaksi käytännön esimerkkiä Novotekin ratkaisuista. Työn lopulla paneudutaan insinöörityöllä ratkaistuun ongelmaan Historian-ohjelmiston ja Azure-pilvipalvelun välisessä tiedonsiirrossa.

Novotek pyrkii lisäämään toimittamansa GE Historian -ohjelmiston käyttöä teollisen internetin ratkaisuissaan tietovaraston asemassa. Historianin uusimmasta ohjelmistoversiosta 7.0 löytyvä REST-rajapinta mahdollistaa tietokannan käyttämisen lähes missä vain pilvipalvelussa, mutta tämä insinöörityö keskittyi Azure-pilvipalveluun Novotekin saaman suuren kysynnän vuoksi. Työssä käydään läpi sekä rajapintaa ja sen tietoturvallista käyttöä että Azure Data Factory -palvelua, jolla tiedonsiirto pilvensisäisesti toteutettiin.

Insinöörityön tuloksilla GE Historian -ohjelmiston sisältävää IoT-ratkaisua voidaan kehittää eteenpäin ja myydä palveluita nykyisille ja tuleville asiakkaille, joilla ohjelmisto on jo asennettu.

Avainsanat	IoT, teollinen internet, tiedonsiirto, integraatio, historiatietokanta

Author	Jani Arponen
Title	Data Transfer for Industrial Internet and the Use of an Operational Historian as Cloud Storage
Number of Pages	47 pages
Date	8 May 2018
Degree	Bachelor of Engineering
Degree Programme	Automation Technology
Professional Major	
Instructors	Jukka Pirinen, IoT Solutions Manager Timo Kasurinen, Senior Lecturer

This study was commissioned by Novotek Oy and concerns data transfer for Industrial Internet. Novotek delivers industrial IT and automation solutions and products based on standardized products and software. The aim of the study was to explore Novotek's data transfer solutions from factory floor to the cloud and to integrate GE Historian as a database to Microsoft Azure cloud platform.

The beginning of this thesis explores data transfer, security and cloud services from a theoretical standpoint and presents two of Novoteks solutions to secure data transfer for Industrial Internet. At the end of the thesis, the integration of GE Historian to Microsoft Azure is explained.

Novotek aims to increase the use of GE Historian as a database in the cloud for its data transfer solutions. A recent major software update added a REST API to Historian and this enables its use as a database in almost any cloud platform, but this study focused on Microsoft's Azure due to it's high demand from Novoteks clients. The thesis explores the REST API and a secure way of using it for integrating Historian to Azure using Azure Data Factory.

The results of this study can be used to further develop IoT solutions using GE Historian and enables Novotek to offer services to current and future clients.

Keywords	IoT, industrial internet, data transfer, integration, operational historian

Sisällys

Lyhenteet

AMQP	Advanced Message Queueing Protocol. Viestijonoihin perustuva tietoliikenneprotokolla.
API	Application Programming Interface. Ohjelmointirajapinta.
CPS	Cyber-physical system. Kyberfyysinen järjestelmä.
CoAP	Constrained Application Protocol. Tietoliikenneprotokolla suorituskyvyltään rajoitetuille laitteille.
DMZ	Demilitarized Zone. Ei-kenenkään-maa-verkkomalli.
HTTP	Hypertext Transfer Protocol. Verkkoselainten käyttämä tiedonsiirtoprotokolla.
HTTPS	HTTP Secure. HTTP-protokollan salattu versio.
IIoT	Industrial Internet of Things. Teollisten esineiden ja asioiden internet, kutsutaan myös teolliseksi internetiksi (industrial internet).
IoT	Internet of Things. Esineiden ja asioiden internet.
JSON	JavaScript Object Notation. JavaScript-ohjelmointikielen mukainen tiedosto- ja tietomuoto.
MQTT	Message Queueing Telemetry Transport. Kevyt, tuottaja-tilaajamallin tietoliikenneprotokolla.
OAuth	Avoin standardi verkkopalveluiden autentikointiin kolmannelle osapuolelle.
REST	Representational State Transfer. HTTP-protokollaan perustuva ohjelmointirajapintamalli.

1 Johdanto

Tämän insinöörityö tehdään Novotek Oy:lle. Yritys toimittaa teollisuuden IT- ja automaatioratkaisuja tiedon keräämiseen, tallentamiseen, analysointiin ja jakamiseen, pääpainopisteenä valmistavan teollisuuden tuotanto. Työssä avataan teollista internetiä kokonaisuutena, pääpaino on teollisen prosessin integroimisessa pilvipalveluun ja teollisen datan siirtämisessä pilven tietovarastoihin.

Insinöörityössä tarkastellaan lähemmin kahta Novotek Oy:n ratkaisua teollisen datan tiedonsiirrosta. Ensimmäisessä ratkaisussa teollinen data siirretään Microsoft Azure -pilvipalvelun IoT Hub- ja Stream Analytics -työkalujen avulla Azuren tietovarastoihin. Toisessa ratkaisussa GE Historian -ohjelmistolla toteutetaan pilvipohjainen historiatietokanta Azureen sekä mahdollistetaan tiedonsiirto Azuren omiin tietovarastoihin ja analytiikkatyökaluille Historianin REST-rajapinnan avulla.

Ongelma, jota insinöörityöllä pyritään ratkaisemaan, on Historian-ohjelmiston REST-rajapinnan käyttäminen Azure-pilvipalvelun sisäiseen tiedonsiirtoon. Yritykselle insinöörityön aihe on ajankohtainen kasvavan IoT-kysynnän vuoksi sekä yrityksen halusta käyttää Historiania IoT-ratkaisuissaan. Työn päätavoitteena on saada tiedonsiirto toimimaan tietoturvallisesti ja luotettavasti. Toissijaisina tavoitteina on tutustua Azure-pilvipalvelun tarjoamiin palveluihin ja infrastruktuuriin sekä Microsoftin Power BI -ohjelmistoon datan visualisoinnissa.

2 Teollinen internet

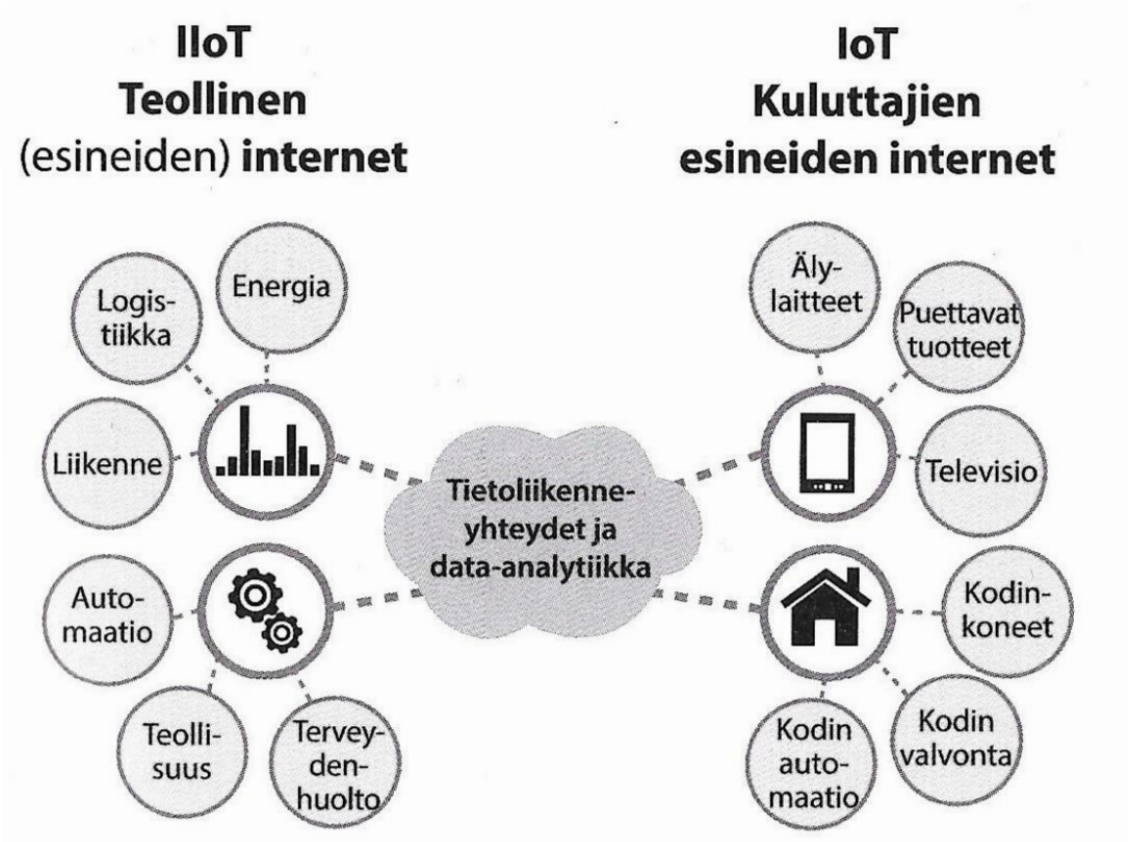
2.1 Esineiden ja asioiden internet

Internet of Things (IoT), esineiden ja asioiden internet, on ollut viime vuosina nopeasti nouseva tekniikka ja ilmiö, joka hyödyntää fyysisten esineiden ja asioiden sensoreilta keräämää dataa verkottamalla nämä internetiin ja saa esineet keskustelemaan keskenään. Sensoriarvoja analysoidaan IoT-laitteella tai internetin pilvipalveluissa ja datasta tuotetaan lisäarvoa yrityksille ja asiakkaille. [1.] IoT sai nimensä jo vuonna 1999 Kevin Ashtonin RFID-tekniikkaa koskevan esityksen yhteydessä, mutta vasta viimeisen kymmenen vuoden aikana ovat eri sensori-, tiedonsiirto- ja data-analytiikkateknologiat kehittyneet ja halventuneet pisteeseen, joka on osaltaan mahdollistanut IoT:n. [2; 3, s. 44.]

IoT:llä on monia määritelmiä, jotka riippuvat paljolti siitä, kuka asiasta puhuu. Yhdysvaltalaisen verkkolaitteita valmistavan Ciscon määritelmän mukaan IoT on se ajanhetki, jolloin internetiin liitettyjen laitteiden määrä nousi suuremmaksi kuin ihmisten. Tämä tapahtui vuosien 2008 ja 2009 välillä, jolloin internetiin oli liitetty yli 7 miljardia laitetta. [4, s. 2.] Moni muu määritelmä keskittyy Internet of Things -nimen määrittelyssä sanaan Thing, eli esineeseen ja niiden yksilölliseen tunnistettavuuteen. IoT:n nimensä mukaisesti käyttämää Internet-tekniikkaa hyödyntäessä tämä käytännössä tarkoittaa IP-osoitetta. Tulevaisuudessa tullaan varmasti näkemään myös IPv6-osoitteiden käyttöä IoT-laitteiden massiivisen määrän vuoksi, jonka on eri arvioiden mukaan ennustettu kasvavan vuoteen 2020 mennessä 20−30 miljardiin laitteeseen. [3, s. 30−32; 5; 6.]

2.2 Teollinen internet

Internet of Things on sivuhaara suuremmasta CPS-kokonaisuudesta (Cyber-Physical Systems), joka kuvaa kyberfyysisiä järjestelmiä. Kyberfyysiset järjestelmät koostuvat laitteista, jotka keskustelevat toistensa kanssa. Toinen sivuhaara, jolla on teknologisesti paljon päällekkäisyyksiä IoT:n kanssa, on Industrial Internet of Things (IIoT) eli teollisten esineiden ja asioiden internet tai lyhyemmin ilmaistuna Industrial Internet eli teollinen internet. Teollinen internet ja IoT ovat tekniikaltaan ja toteutustavoiltaan hyvin samantapaisia ja voidaankin sanoa, että ne eroavat lähinnä käyttökohteensa (kuva 1) puolesta toisistaan. [3, s. 29−31; 7, s.71.]



Kuva 1. Teollisen ja kuluttajien IoT:n jakautuminen käyttökohteen mukaan [3, s.31].

Termi teollinen internet on yhdysvaltalaisen General Electricin (GE) vuonna 2012 keksimä nimitys. Vaikka nimi viittaakin perinteiseen valmistavaan teollisuuteen, sisältää termi monia muitakin aloja, kuten terveydenhuollon ja liikenteen. Teollinen internet pitää sisällään valmistavan teollisuuden lisäksi muun muassa energian tuotannon voimalaitoksista älykkäisiin sähköverkkoihin, liikenteen ja itseajavat autot, terveydenhuollon, julkisen infrastruktuurin kuten vesi- ja jätehuollon sekä monia muita aloja. [3, s. 29−31; 8.] Tässä insinöörityössä on pyritty tekstissä erottelemaan kaupallinen ja teollinen IoT käyttämällä kaupallisesta puolesta termiä IoT ja teollisesta puolesta termiä teollinen internet.

Teollisen internetin määritelmä yleisesti ja virheellisesti yhdistetään Teollisuus 4.0 -käsitteen synonyymiksi. Automaation uudeksi, neljänneksi vallankumoukseksi povattu Teollisuus 4.0 -ilmiö on Saksan hallituksen vuonna 2012 julkistama strateginen Industrie 4.0 -hanke. Hankkeen tarkoituksena on varmistaa Saksan kilpailukyvyn säilyminen tulevaisuuden teollisuudessa hyödyntämällä monia digitalisaation ja data-analytiikan tuomia hyötyjä. Teollisuus 4.0 -mallissa tehtaat luodaan kyberfyysisistä ratkaisuista ja järjestelmistä sekä älykkäistä koneista, jotka kommunikoivat toisten älykkäiden koneiden ja

järjestelmien kanssa, muodostaen älykkään tehtaan. Tämän mallin tehtaiden ja koko hankkeen toteuttamiseen tarvitaan teollisen internetin tekniikoita ja ratkaisuja. [3, s. 37−40.]

2.3 Teollisen internetin hyödyntäminen

Teollista internetiä soveltamalla voidaan mahdollistaa muun muassa etävalvontaa ja etähallintaa teollisuuden laitteille, prosesseille ja koko tehtaille. Tällaiset sovellukset ovat olleet arkipäivää valmistavassa teollisuudessa jo yli 20 vuotta erilaisten valvomo- ja SCADA-järjestelmien avulla, mutta vanhoissa valvomototeutuksissa prosessin arvoihin on suhtauduttu lähinnä prosessin valvonnan ja ohjauksen näkökulmista. Valvomon ruudulla näkyvän tilanteen mukaan suoritetaan jokin säätötoimenpide tai verrataan vanhaan prosessiarvoon. Teollinen internet tuo kuvaan mukaan pilvipalvelut, data-analytiikan työkaluja sekä koneoppimista tähän vanhaan keksintöön, jolloin prosessin arvoista muodostuu raaka-ainetta, dataa. Datasta jalostetaan analyysin keinoin informaatiota ja tietoa prosessista, joiden avulla prosessia ja tuotantoa voidaan optimoida jopa etänä. [3, s. 48−50, 61−62.]

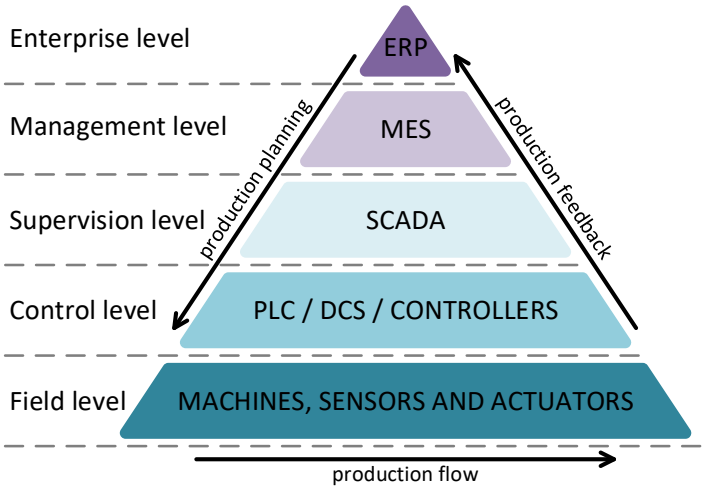
Prosessista kerättävien ajasta riippuvien sensoriarvojen analysoinnilla pilvipalveluissa ja mahdollisilla koneoppimisalgoritmeilla voidaan mahdollistaa etänä suoritettava kunnonvalvonta sekä ennakoiva huolto. Klassinen esimerkki tästä on IoT-tekniikkaa hyödyntävä värähtelymittaus koneesta. Mitattavien arvojen vaihtelulla pyritään ennustamaan laiterikkoja ennen kuin ne tapahtuvat ja näin säästetään tuntuvasti huollon kustannuksissa ja mahdollisista prosessin keskeytyksistä syntyvät tappiot. Teollinen internet mahdollistaa myös täysin uudenlaiset data- ja tietopohjaiset liiketoiminnat. Laitevalmistaja voi kerätä metatietoa laitteestaan ja mahdollistaa tämän avulla erillisten laitteiden etäkunnonvalvonnan ja ennakoivan huollon sekä palvelupohjaisen liiketoimintamallin, jossa laite myydään palveluna ja asiakkaalta laskutettava hinta määräytyy metatiedon, kuten käyttötuntien tai tuotannon määrän mukaan. [3, s. 73−83.]

Teollinen internet kulminoituu Teollisuus 4.0 -mallin kaltaisiin älykkäisiin tehtaisiin, älykaupunkeihin ja -infrastruktuureihin. Älykkäissä tehtaissa kaikki laitteet hyödyntävät teollista internetiä prosessitasolta logistiikkaan. Tehdas osaa itse tilata, varastoida ja hallita raaka-aineita, joista se eri prosesseillaan valmistaa aikataulun ja kaupallisen tuoton

mukaan optimoituja tuotteita, pakkaa ja varastoi ne sekä osallistuu jopa myyntiprosessiin. Nämä ovat tulevaisuuden teollisen internetin sovelluksia. [3, s. 61, 86−90.]

2.4 Teollisen internetin sovelluksia

Novotek Oy keskittyy valmistavaan teollisuuteen ja tarjoaa muun muassa tuotannonohjaukseen ja -raportointiin soveltuvia ratkaisuja. IEC 62264 / ISA 95 -standardin mukaisen automaation tasoja kuvaavan pyramidin (kuva 2) mallissa tuotannonohjaus ja -raportointi sisältyvät MES-tasoon (Manufacturing Execution Systems). MES-järjestelmillä ohjataan ja suunnitellaan tuotantoa ja tuotantoerien ajoittamista. Tulevaisuudessa MES-sovellukset löytyvät yhä useammin pilvestä ja tuotannon raportoinnin siirtäminen pilveen on hyvä lähtökohta yrityksen teollisen internetin strategian aloittamiseen. [9; 10, s. 12.]



Kuva 2. ISA 95 -standardin määrittelemä automaation tasojen pyramidi [10, s. 12 mukailtu].

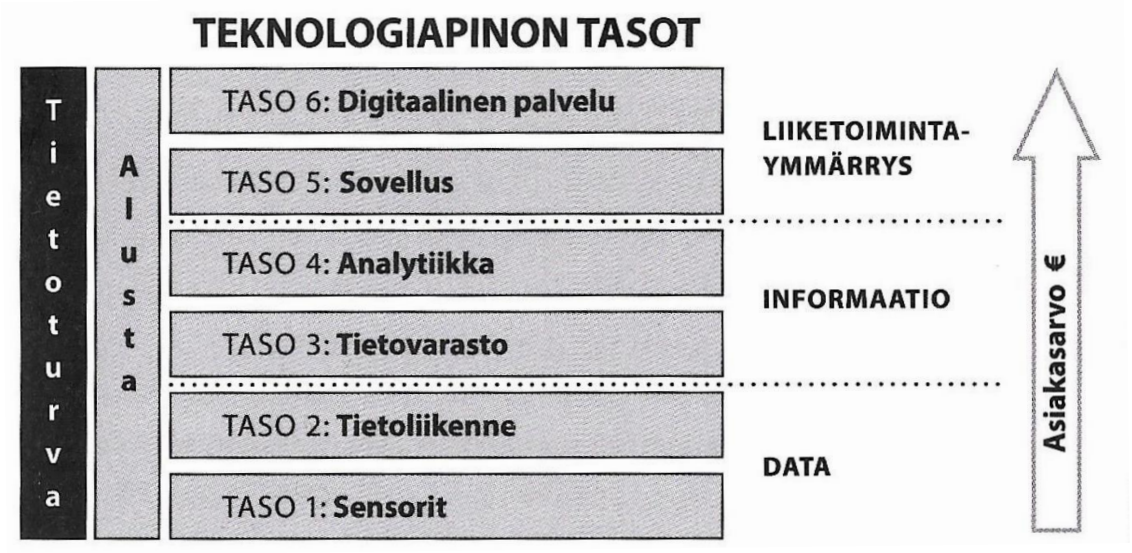
Teollista internetiä hyödyntävä tuotannon raportointi -järjestelmä kerää dataa yhdeltä tai useammalta tehtaalta ja näiden prosesseista, aggregoi datan yhteen pilvipalveluun, jossa datasta jalostetaan informaatiota ja tietoa tuotannon menneestä, nykyisestä ja tulevasta tilasta. Pilvipalvelun näytöillä voidaan näyttää reaaliaikatietoja kuten tuotantoerien suoritusaikoja sekä koneiden ja prosessien OEE-lukuja (Overall Equipment Effectiveness) eli käytettävyyttä, toiminta-astetta ja tuotannon laatukerrointa. Lisäksi voidaan luoda raportteja tuotannonohjaukselle, joista pystyy näkemään esimerkiksi prosessien puollonkauloja, kehittämään kunnossapidon toimintoja ja optimoimaan prosessia. Raportteja voidaan verrata toisiinsa automaattisesti ja muodostaa eräraporttien ryhmiä eri resepteistä ja verrata näin laitteiston toimintaa eri resepteillä. [9.]

3 Tiedonsiirto pilveen

Kaikki teollisen internetin sovellukset lähtevät liikkeelle käytännössä samasta pisteestä. Teolliseen prosessiin tai laitteeseen on pystyttävä liittymään ja siltä haluttava data on pystyttävä siirtämään internetin yli johonkin pilvipalveluun ja tallentaa tietokantaan. Kaikki tämä on pystyttävä toteuttamaan tietoturvallisesti ja luotettavasti.

3.1 Teollisen internetin arkkitehtuuri

Yleisellä tasolla teollisen internetin arkkitehtuuri voidaan kuvata koostuvan kuvan 3 mukaisesta teknologiapinosta. Teknologiapino pilkkoo arkkitehtuurin teknisesti erilaisiin irrallisiin osakokonaisuuksiin, jotka ovat erikseen toteutettavissa. [3, s. 142−143.] Teknologiapinoja on seuraavaksi pyritty avaamaan geneerisellä tasolla teollisen prosessin näkökulmasta. Tämä insinöörityö keskittyy tasoille kaksi ja kolme sekä tason neljä mahdollistamiseen.



Kuva 3. Teollisen internetin teknologiapino [3, s. 143].

Teknologiapinon pohjalla ovat sensorit, joita teollisessa prosessissa voidaan pitää yleisnimityksenä kaikille laitteille, kuten verkotetuille antureille, toimilaitteille, logiikoille, taajuusmuuttajille ja teollisuustietokoneille, joilta voidaan lukea prosessin fyysisten ilmiöiden luomaa dataa. Sensorit ovat koko teollisen internetiä hyödyntävän järjestelmän perusta; ilman sensoreita ei ole dataa, ilman dataa ei ole teollista internetiä. [3, s. 142−143, 152.]

Teknologiapinon toisella tasolla sensoriarvot eli data siirretään tietoliikennetekniikan keinoin analyysiä varten ja mahdollistetaan prosessin ja järjestelmien etäkäyttö ja -hallinta. Tietoliikennetaso koostuu verkkolaitteiden ja erilaisten verkkoratkaisujen lisäksi viestinnän protokollista ja standardeista. Teollisen internetin käyttötarkoituksiin sopivia tietoliikenneprotokollia on olemassa useita, ja näistä osaa on pyritty kuvaamaan tarkemmin luvussa 3.3. Kuvan 3 tietoliikennetasoon on myös sisällytetty teolliseen prosessiin liittyminen. [3, s. 142−143, 163−164.]

Tasot 3−6 voisi myös kuvata yhdellä, laajemmalla tietotekniikka-nimisellä tasolla, koska ne kaikki käyttävät tietotekniikan välineitä prosessista siirretyn datan hyödyntämisessä ja sen jalostamisessa informaatioksi sekä tiedoksi. Kolmas eli tietovarastotaso kuvaa teollisen datan, informaation ja/tai tiedon tallentamista keskitettyyn, yleensä pilvipohjaiseen tietokantaan, johon voidaan helposti lisätä muita prosessia koskevien tietolähteiden, kuten yrityksen tuotannon- ja toiminnanohjausjärjestelmien tuottamia ja käyttämiä arvoja. Teollisen prosessin sensorien tuottama data on ajasta riippuvaa, ja sitä voi syntyä massiivisia määriä. Keskivertotehtaassa on sadoista tuhansiin mittapistettä, näihin lisättynä säätöpiirien asetukset, raja-arvot ja käyttöparametrit sekä hälytykset, jonka jälkeen päästään jopa kymmeniin tuhansiin mittauspisteisiin, joita tietokantaan mahdollisesti kirjoitetaan. Tietokannan on suuren data- ja tietomäärän vuoksi oltava helposti skaalautuva ja rakenteeltaan mahdollistettava lähes kaikenlaisen ja -muotoisen datan tallentamisen, vaikka teollisen prosessin tuottama data onkin melko strukturoitua. Pitää myös muistaa, että tietokanta ei ole vain varastointia varten, vaan sieltä on pystyttävä myös noutamaan suurikin määrä dataa nopeasti. [3, s. 143, 195−200; 9.]

Teknologiapinon analytiikkatasolla pilvipalveluun siirrettyä dataa analysoidaan ja siitä jalostetaan informaatiota ja tietoa, joiden perusteella voidaan tehdä liiketoimintaa tukevia johtopäätöksiä. Analytiikan oletuksena on, että tiedetään, mitä datasta halutaan etsiä. Tämä päättää, missä muodossa data tuodaan analytiikkaa varten, mitä työkaluja sekä teknologioita analytiikassa käytetään ja päättää myös tuotetun informaation ja tiedon visualisoinnista. Mikäli teollisesta prosessista pyritään muodostamaan eräraportteja ja tuotannonohjausta kiinnostavaa informaatiota, koneoppiminen ei välttämättä ole paras työkalu analyysiin. Jos päämääränä onkin laitteiden ennakoiva huolto erilaisten poikkeamien ja historiallisten virhetilanteiden perusteella ennustaminen, voi koneoppiminen ja neuroverkot olla varteenotettava ratkaisu. Teollisen internetin data-analytiikka voidaan ajatella jakautuvan kuvailevaan, diagnostiseen, ennakoivaan ja ohjaavaan analyysiin.

Kuvaileva analytiikka kertoo, mitä prosessissa tapahtui, ja sen avulla voidaan tuottaa esimerkiksi edellä mainittuja eräraportteja tuotannonohjaukseen. Diagnostinen analytiikka kertoo, miksi ja miten jotain tapahtui. Tällä tasolla hyödynnetään tiedonlouhintaa ja tilastollista analyysiä. Ennakoiva ja ohjaava analytiikka kertovat tulevaisuudesta, mitä prosessille tapahtuu ja miten saadaan haluttu lopputulos aikaan. Tällä analytiikan tasolla luotetaan koneoppimiseen ja simulointiin sekä optimoidaan prosessia erilaisten analyysien tulosten perusteella ja ennakoidaan laitteiden huoltotarvetta. [3, s. 143, 205−209.]

Viimeiset kaksi numeroitua tasoa kuvassa 3, sovellus- ja digitaalisen palvelun -tasot ovat tämän insinöörityön rajojen ulkopuolella, mutta lyhyesti sanottuna näillä tasoilla sensorien mittauksista saatavaa dataa, josta on analytiikan keinoin jalostettu informaatiota ja tietoa, hyödynnetään yrityksen tai sen asiakkaan liiketoiminnassa tuottamalla erilaisia sovelluksia, kuten edellä mainittuja eräraportteja [3, s. 217−219].

Kaikkia teknologiapinon tasoja yhdistää alusta. Alustan tehtävänä on yhdistää eri teknologiapinon osia yhdeksi tasoksi, käytännössä tasolta kolme ylöspäin, esimerkiksi aggregoimalla eri lähteistä virtaava data yhteen tietovarastoon, mahdollistaa tämän datan analysointi sekä sovelluksien rakentaminen analysoidun datan ympärille. IoT:n ja teollisen internetin sovelluksissa alustana toimivat internetin pilvipalvelut. IoT:n ja teollisen internetin näkökulmista hyvän pilvialustan tulisi mahdollistaa

- alustaan liittymisen usealla eri tietoliikenneprotokollalla
- laitehallinta kenttälaitteille
- erilaiset tietokannat erimuotoiselle datalle, jotka skaalautuvat helposti tietomäärien kasvaessa
- tapahtumapohjaisten sääntöjen määrittely sekä toimintojen tekeminen ja hallinta tapahtumien mukaan
- datan analysointi erilaisilla analytiikkatyökaluilla
- analysoidun ja raa’an datan visualisointi
- rajapinnat ulkoisiin tietojärjestelmiin kuten muihin pilvipalveluihin ja yritysten toiminnanohjausjärjestelmiin
- muita työkaluja esimerkiksi raportointiin ja sovelluskehitykseen. [3, s. 227−233; 11, s. 7−9.]

3.2 Teollisen internetin tietoturva

Perinteisesti teollisuuden suhtautuminen tietoturvaan on ollut hyvä. On ymmärretty, että tietoturva on erittäin tärkeää, mutta paljolti on luotettu vääriin olettamuksiin, kuten siihen, että teollisuuden järjestelmät ovat automaattisesti tietoturvallisia, koska ne ovat täysin tai palomuurein erotettuja internetistä tai siihen, että mahdolliset hyökkääjät eivät ymmärrä teollisista prosesseista eivätkä halua hyökätä tehtaisiin. Nämä olettamukset ovat nykymaailmassa vääriä. [3, s. 241−243.] Shodan.io-hakupalvelun avulla voidaan löytää internetiin liitettyjä eri tasoin suojaamattomia laitteita, myös teollisuus- ja IoT-laitteita. Pelkästään automaatiossa paljon käytetyn Modbus-protokollan suojaamattomia laitteita hakukone löytää maailmanlaajuisesti 15 565, joista 84 kappaletta Suomessa (haku suoritettu 19.3.2018). Myönnettäköön, että Modbus-protokollassa ei ole sisäänrakennettua tietoturvaa, mutta tämä on todellisuutta myös monessa muussa teollisuudessa käytössä olevassa protokollassa. Vaikka Modbus-protokollaa käyttävä järjestelmä olisi suojattu erottamalla se internetistä palomuurilla, siirtyy haavoittuvuus vain astetta ylemmäs palomuurin takana olevalle palvelimelle, jolla on oikeus keskustella Modbus-järjestelmän kanssa. Tämän palvelimen tai tietokoneen tietoturvan määrittelee muun muassa sen käyttöjärjestelmä, sen ikä ja tietoturvapäivitysten asentamistaajuus, joka monessa teollisessa ympäristössä on valitettavan suuri tai jopa olematon. [12; 13.]

3.2.1 IoT-laitteiden tietoturvariskit

IoT-laitteet ovat valitettavan usein täysin vailla tietoturvaa. Mahdollisia syitä tälle voi olla monia, mutta usein syynä on yksinkertaisesti kustannus. Kuluttajille suunnattujen halpojen IoT-laitteiden kehityksessä ei kannata panostaa tietoturvaan, koska keskimääräinen kuluttaja ei välttämättä ole edes tietoinen kaikista mahdollisista riskeistä ja aukoista, joita IoT tuo mukanaan. Esimerkkinä voidaan ottaa vuoden 2016 lopulla tehdyt palvelunestohyökkäykset Mirai-botnetiksi nimetyllä tietoturvattomista ja saastuneista IoT-laitteista muodostuneella bottiverkolla. Palvelunestohyökkäykset olivat historian siihenastisista suurimmat ja onnistuivat saamaan internetin peruspilareina toimivia DNS-palvelimia polvilleen. Mirai-botnet muodostui arvioiden mukaan noin 100 000 päätepisteestä, suurimmaksi osaksi suojaamattomista IoT-laitteista. [14, s. 49; 15.] Tietoturvan puuttumisen esimerkkiä voidaan ottaa myös pienemmässä mittakaavassa sekä kotimaan rajojen sisältä. Vuoden 2016 marraskuussa hakkerit pääsivät lämmitys- ja jäähdytysjärjestelmiin, siis teollisen internetin järjestelmiin ja aiheuttivat Rauman jäähallin jään sulamisen ja katkaisivat talojen lämmityksiä Lappeenrannassa [16]. Nämä esimerkkitapahtumat ovat

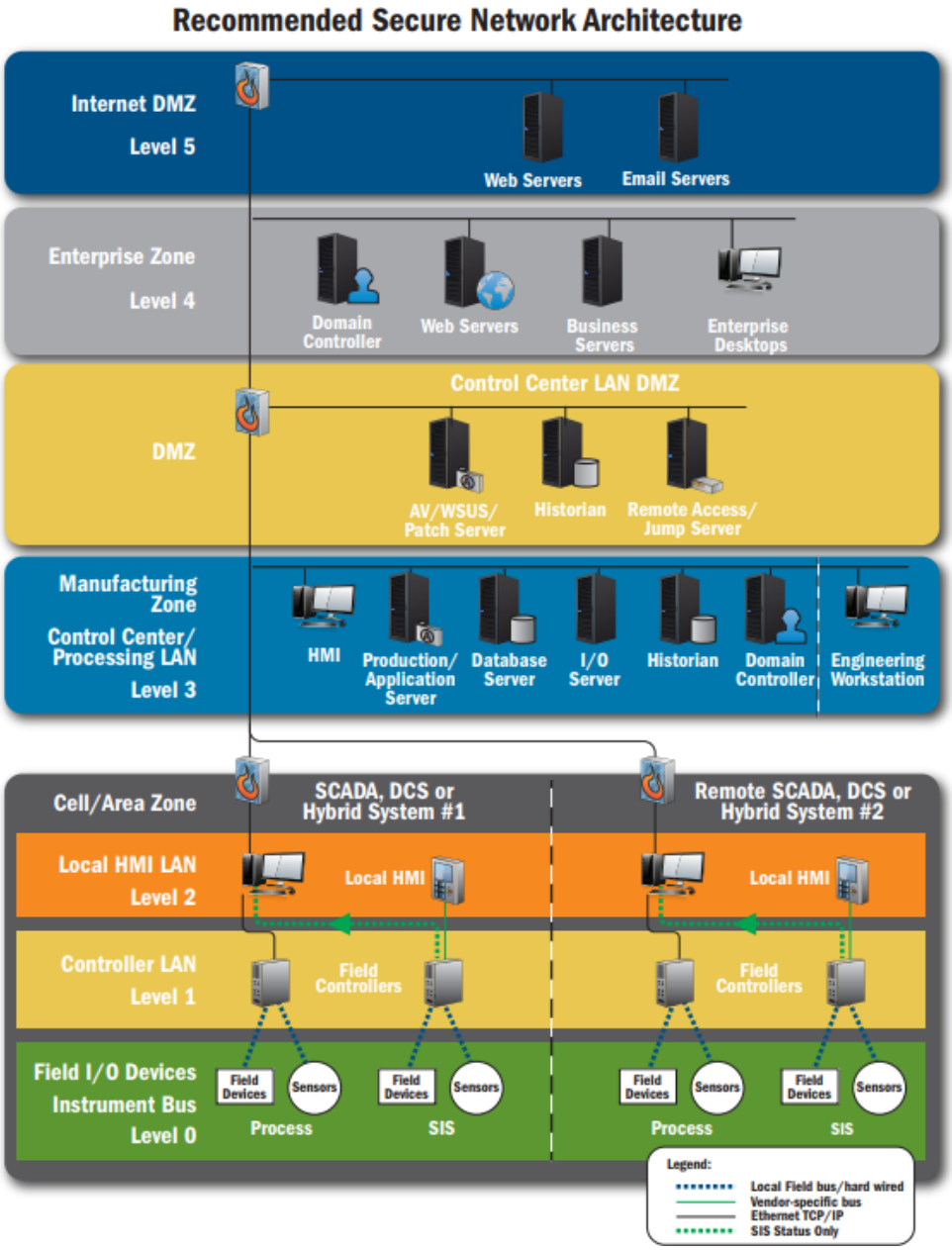
pientä verrattuna teollisen internetin tietoturvan pettämiseen infrastruktuurisesti kriittisissä järjestelmissä, kuten energiantuotannossa ja sähköverkoissa tai ihmishenkiä uhkaavat tietoturvamurrot lentoliikenteeseen tai öljyvarastoihin [3, s. 243−245].

3.2.2 Tietoturvaratkaisuja

Uusia teollisen internetin ja IoT:n sovelluksia ja ratkaisuja luodessa tietoturva pitää olla kunnossa ja mukana koko sovelluksen elinkaaren ajan aivan suunnitteluprosessin alusta ylläpitoon sekä itse prosessin että sen tuottaman datan näkökulmista. Jo olemassa olevaan teolliseen prosessiin liityttäessä tulisi tehtaan verkkotopologia sekä sen sisältämät laitteet ja järjestelmät ja niiden muodostamat yhteydet dokumentoida yksityiskohtaisesti sekä luoda tietoturvan riskianalyysi. Valitettavan usein kenelläkään tehtaalla ei ole kaikenkattavaa kuvaa koko verkosta ja tietojärjestelmistä, joka vaikeuttaa dokumentointia. Uusien yhteyksien luomisessa pitää noudattaa tiukkoja tietoturvasääntöjä, vain valtuutetut ja tunnistautuneet käyttäjät ja yhteydet sallitaan. Tulee myös pohtia, tarvitaanko kaksisuuntaisia verkkoyhteyksiä ollenkaan vai onko teollisen datan siirtäminen pilveen riittävä ratkaisu yritykselle. Sulkemalla tien avaus pilvestä prosessille luovutaan joistain teollisen internetin tuomista mahdollisuuksista kuten etähallinnasta ja etäoptimoinnista. Palomuurien porttiasetuksien määrittäminen sekä yhteyden avaamisen sallivien osoitteiden hallinta ovat arkipäivää IT-osastoille, mutta operatiivisen toiminnan henkilöstölle nämä voivat olla uusia asioita, joten on tärkeää, että IoT-projekteja tehtäessä sekä IT- että OT-henkilöstö ovat mukana projektissa. Erittäin tärkeissä osissa prosessia tulee myös pohtia, onko verkottaminen edes vaihtoehto vai pidättäydytäänkö vanhassa teollisuuden tavassa erottaa kriittisimmät osaprosessit täysin muusta verkosta. Prosessin arvoja voidaan myös lukea prosessista irrallaan olevilla antureilla, jotka eivät missään vaiheessa kytkeydy automaatioverkkoon vaan toimivat irrallisena järjestelmänään. Tämä ratkaisu yhdistettynä muun vahvan tietoturvan kanssa mahdollistaa erittäin tietoturvallisen teollisen internetin sovelluksen. [3, s. 245−247; 9.]

Yksi suosituimmista, tietoturvallisista tavoista liittää teollinen prosessi yrityksen verkkoon ja myös ulkoverkkoon, on niin kutsuttu ei-kenenkään-maa- eli DMZ-malli (Demilitarized Zone) (kuva 4). DMZ-mallissa luodaan fyysinen ja looginen aliverkko, joka toimii turvallisena yhdyskäytävänä epäluotettavan verkon, yleensä internetin, ja turvattavan verkon välillä. Automaatioverkot erotetaan yleensä myös yrityksen toimistoverkosta DMZ-mallin avulla. Toimistoverkosta automaatioverkkoon ei ole suoraa yhteyttä, vaan DMZ-alueelle sijoitetaan laitteita, jotka mahdollistavat pääsyn verkkojen välille. Hyvä esimerkki tästä

on automaation historiatietokannan sijoittaminen DMZ-alueelle, jolloin automaatiolaitteilta kerättävät sensoriarvot ovat luettavissa yrityksen verkosta. DMZ-alueen käyttäminen teollisessa internetissä toimii hyvin samoin tavoin, mutta epäluotettavana verkkona toimii internet. Kaikkia tietoturvaongelmia DMZ-malli ei kuitenkaan ratkaise. Alueen palvelimet ovat alttiina verkkohyökkäyksille, mutta tätä riskiä voidaan pienentää palvelinten päivitysten ja yleisen tietoturvan ylläpidolla ja hallinnalla. Vaikka hyökkäys DMZ-alueelle onnistuisikin, tiukoilla tietoturva-asetuksilla alueiden välillä varmistetaan, ettei hyökkäys pääse etenemään automaatio- tai yritysverkkoihin. [17, s. 16−20.]



Kuva 4. Yhdysvaltain Kotimaan turvallisuus -viraston suosittelema DMZ-malli teollisten verkkojen tietoturvaan [17, s. 17].

3.3 Teolliseen prosessiin liittyminen ja integraatio

Teollista internetiä kuvaavan teknologiapinon (kuva 3) mukaan sensoriarvot siirretään tietoliikennetekniikoilla tietovarastoon, mutta rinnastaessa tämä teolliseen prosessiin tai tehtaaseen huomataan, että tämä kirjaimellisesti tarkoittaisi verkotettuja tai älykkäitä sensoreita, jotka syöttäisivät dataa suoraan pilveen automaatiojärjestelmän lisäksi. Tehtaat ovat kuitenkin hyvin harvoin, jos koskaan toteutettu tällaisilla teknologioilla. Yleisesti teollisen prosessin sensorit, kuten erilaiset lähestymisanturit ja rajakytkimet on yhdistetty mahdollisten etä-IO-laitteiden avulla tai suoraan PLC-laitteille (Programmable Logic Controller) tai suurempiin DCS-automaatiojärjestelmiin (Distributed Control System). Sensorien arvot pitää lukea näiltä järjestelmiltä. PLC-ohjelma lukee arvoja muistiinsa laitevalmistajasta riippuen erilaisiin muuttujalistoihin tai muihin tietorakenteisiin ja suorittaa näiden perusteella ohjelmaa, joka ohjaa prosessia. PLC:t ja muut teollisuuden laitteet, kuten taajuusmuuttajat ja operointipaneelit, keskustelevat keskenään jotain laitevalmistajien määrittämiä protokollia, joiden avulla laitteita voidaan liittää toisiinsa.

3.3.1 OPC-standardi

Automaatiojärjestelmien, jotka sisältävät usean eri laitevalmistajan laitteita, ja niiden käyttämien protokollien integroiminen toisiinsa erilaisilla ajureilla ja protokollamuunnoksilla on ollut arkipäivää teollisuudessa jo vuosikymmeniä. Historiallisesti uusien laitteiden ja laitevalmistajien tuloa markkinoille hidasti muun muassa ajuriohjelmistojen luonti jokaiselle mahdolliselle protokollakombinaatiolle eri laitevalmistajien kesken. Tätä ja muita laiteintegraation ongelmia ratkaisemaan teollisuuden suuret laitevalmistajat perustivat OPC-säätiön vuonna 1996, johon tänään kuuluu yli 450 jäsentä. OPC-säätiön päätehtävänä on ylläpitää OPC-standardia, joka mahdollistaa säätiöön kuuluvien laitevalmistajien luomien protokollien helpon, turvallisen ja luotettavan integraation teollisessa automaatiossa. OPC-lyhenne muodostui aluksi sanoista Object Linking and Embedding (OLE) for Process Control, mutta nykyään lyhenne muodostuu sanoista Open Platform Communications. [18.]

OPC-standardi syntyi OPC-säätiön työnä vuonna 1996 ja perustui Microsoftin COM- ja DCOM-teknologioihin ja tarkoituksena oli luoda reaaliaikainen rajapinta PLC-laitteiden ja HMI- sekä SCADA-ratkaisujen väliin niin sanottuna väliohjelmistona. Standardi on kehittynyt vuosien saatossa paljon, luoden eri spesifikaatioita käyttökohteesta riippuen, kuten teollisuudessa eniten käytetty OPC DA (Data Access), hälytys ja tapahtumat

mahdollistava OPC AE (Alarms & Events), historiallisen aikasarjadatan käsittelyyn keskittyvä OPC HDA (Historical Data Access) sekä uusin alustasta riippumaton, kaikki aikaisemmat, nykyään OPC Classic -nimisen spesifikaatiokokoelman sisältävä OPC UA (Unified Architecture). [19.]

OPC-standardin mukaisessa kommunikaatiossa teollisuuden laitteille liitytään niiden omilla protokollilla, joille OPC-ohjelmistoon on sisäänrakennetut ajurit tai rajapinnat. OPC-ohjelmisto kyselee eli pollaa asetetun aikavälein teolliselta laitteelta tämän muuttujien tilatietoja tai arvoja, joka vastaa kyselyyn lähettämällä kysytyt arvot. Mallia kutsutaan kysely-, pollaus- tai asiakas-palvelinmalliksi. Muut OPC-asiakkaat, esimerkiksi SCADA-järjestelmä voivat nyt kysyä OPC-palvelimelta muiden laitteiden muuttujien tiloja, esimerkiksi kokoonpanolinjaston käyntitietoa ja esittää tämän SCADA-valvomon näytöllä. OPC-palvelinta käytetään integraatiomahdollisuuksiensa vuoksi myös tietolähteiden yhdistäjänä. Tämä esimerkki onkin yksi yleisimmistä OPC DA -spesifikaation käyttökohteista. [18.]

Vanhat OPC Classic -spesifikaatiot perustuvat Microsoftin COM- ja DCOM-tekniikoihin ja ovat täten täysin riippuvaisia Windows-käyttöjärjestelmästä. Tekniikoiden tietoturvallisuusongelmien vuoksi OPC:n käyttäminen rajoittui lähinnä automaatioverkon sisäpuolelle, kunnes OPC-säätiö vuonna 2008 julkaisi uuden OPC UA -spesifikaation, joka on erotettu COM- ja DCOM-tekniikoista ja täten täysin alustasta riippumaton standardi. OPC UA toi mukanaan myös parannetun tietoturvan ja tiedonsiirron kryptauksen sekä monia uusia toiminnallisuuksia, kuten hierarkkisen tietomallin, tuottaja-tilaajamallin tiedonsiirron sekä metodien suorittamisen ja monia muita uudistuksia. [19.] OPC UA:n käyttämisestä teollisen internetin tiedonsiirrossa prosessilta pilveen on kerrottu lisää luvussa 3.4.6.

3.3.2 SCADA

Teolliseen prosessiin voidaan siis liittyä erilaisten ajuri- ja OPC-ohjelmistojen avulla, lukemalla automaatiojärjestelmän muuttujien arvoja, jotka kuvastavat järjestelmään liitettyjen sensorien ja toimilaitteiden kautta prosessin fysikaalisia arvoja. Prosessien fysikaaliset arvot ovat kuitenkin vain osa nykyaikaisen PLC- ja SCADA-pohjaisen automaatiojärjestelmän kokonaisuudesta (kuva 2). Teollista prosessia valvotaan ja hallitaan SCADA-valvomojärjestelmällä (Supervisory Control and Data Acquisition). Myös SCADAn hälytystiedot, käyttäjäsyötteet ja muut arvot ovat teollisen internetin näkökulmasta hyödyllistä

dataa, jonka siirtäminen pilveen analyysiä varten kannattaa. SCADA-järjestelmiin pystytään yleisesti liittymään OPC:n avulla. Myös tehtaan tuotannonohjausjärjestelmien arvoja voidaan liittää teollisen internetin piiriin, jolloin esimerkiksi valmistavan teollisuuden eräraportteja voidaan tuottaa pilvessä. [9; 18.]

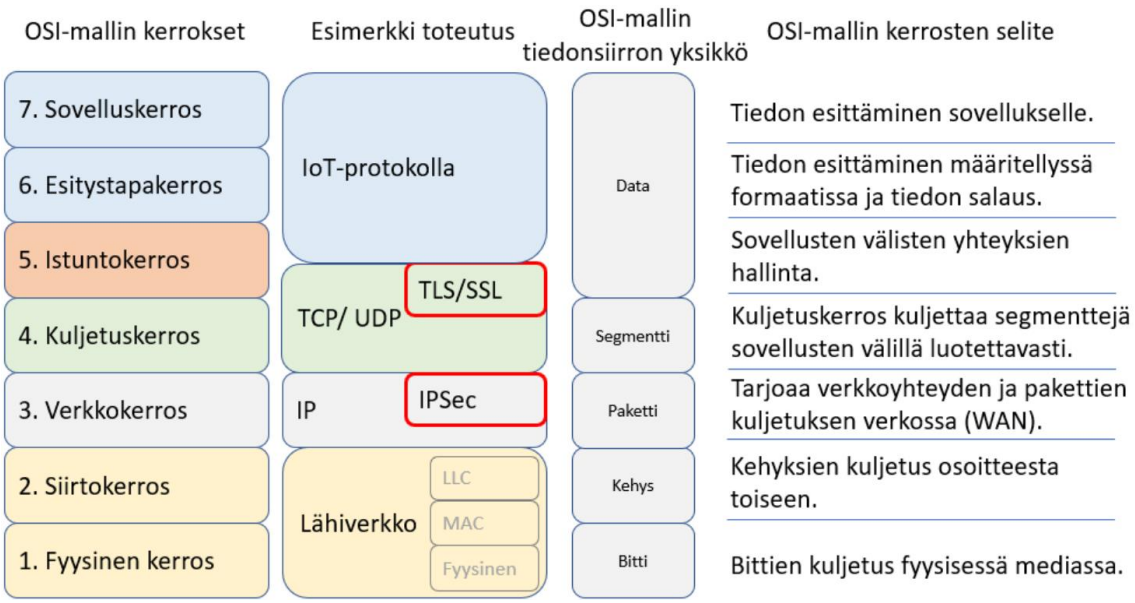
3.4 Teollisen internetin tietoliikenneprotokollat

Teollisen internetin ja IoT:n käyttötarkoituksiin sopivia tietoliikenneprotokollia on monia. Tässä kappaleessa niistä osa käydään läpi pintapuolisesti. Tässä insinöörityössä ei oteta kantaa tietoliikenneyhteyksien langalliseen tai langattomaan toteutukseen sekä keskitytään IP-tekniikan käyttöön.

3.4.1 Yleistä

IoT:n ja teollisen internetin tietoliikenne siirretään jonkin tietoliikenneprotokollan avulla laitteelta toiselle ja/tai pilveen. IoT-laitteet ovat usein prosessointikapasiteetiltaan rajoitettuja pienen kokonsa ja akkukäyttöisyytensä vuoksi, joten käytettävän protokollan on oltava kevytrakenteinen. Myös teollisen internetin sovelluksissa protokollan kevyt rakenne on hyödyksi massiivisten tietomäärien vuoksi, vaikka prosessointikapasiteetti onkin huomattavasti suurempi teollisuus-PC-ratkaisuilla kuin kuluttajille suunnatuissa IoT-laitteissa. Tietoliikenteen pitäisi myös olla salattavissa ja siirto pitää olla luotettavaa. [20, s. 14; 20 s. 125−126.]

Tietoliikennettä kuvataan yleensä OSI-mallin (Open Systems Interconnection Reference Model) avulla (kuva 5). OSI-mallin kerroksilla kuvataan tietoliikenteen koostuminen eri osista fyysisestä välitysmediasta tietoliikenneprotokollaan. Suurin osa IoT- ja teollisen internetin protokollista kulkee OSI-mallin kerroksilla 5−7 kuljetuskerroksen päällä. IoT-sovelluksissa kuljetuskerros toteutetaan yleensä TCP-yhteydellä, mutta myös UDP-yhteyttä käytetään. TCP-yhteyttä (Transmission Control Protocol) käytettäessä muodostetaan pysyvä yhteys päätepisteiden välille ja protokolla varmistaa pakettien saapumisen perille. UDP (User Datagram Protocol) on yhteydetön ja siirron toteutumisesta ei ole takeita ja täten epäluotettava, josta protokolla on saanut liikanimensä Unreliable Datagram Protocol. TCP/IP-kombinaatiosta on muodostunut de facto -standardi verkkoliikenteelle myös teollisuudessa. [20, s. 14−15; 21, s. 81.]



Kuva 5. Rami Ojala kuvaa insinöörityössään ”MQTT IoT-protokolla Toiminta ja toteutus” IoT-protokollien OSI-mallia [20, s. 14].

Kaksi yleistä mallia tiedonsiirtoon eri protokollilla on kyselymalli sekä tuottaja-tilaaja-malli. Kyselymalli (polling) perustuu asiakas-palvelinarkkitehtuuriin (client-server), jossa asiakas kysyy palvelimelta ja palvelin vastaa. Kyselymallia noudattavat esimerkiksi teollisessa automaatiossa suuressa asemassa oleva OPC DA, UDP-yhteyttä hyödyntävä CoAP sekä verkkoselainten HTTP. Tuottaja-tilaajamallissa (publish-subscribe, PubSub) tuottajajäsen siirtää jatkuvasti dataa joko suoraan tai välittäjän (broker) kautta tilaajalle. Tuottaja-tilaajamallin etuna IoT:n näkökulmasta voidaan pitää sen parempaa toimivuutta usean laitteen samanaikaiseen liittämiseen pilvipalveluun sekä tietoliikenteen oleellisella vähenemisellä, kun tilaaja saa välittäjältä vain haluamansa datan. Tuottaja-tilaajamallin protokollia ovat esimerkiksi MQTT ja AMQP. Myös OPC UA tukee tuottaja-tilaajamallia. [3, s. 184−185.]

3.4.2 CoAP

Constrained Application Protocol (CoAP) on suunniteltu nimensä mukaisesti rajoitettujen sovelluksien protokollaksi. CoAP on ideaalinen laitteilla, joilla on erittäin pieni prosessointiteho ja vaativat pienen virrankäytön. Protokolla onkin alun perin suunniteltu pienten laitteiden M2M-kommunikointiin. CoAP on UDP-yhteyden päällä kulkeva, kyselymallin ja client-server-arkkitehtuurin mukainen protokolla, joka muistuttaa paljolti HTTP:tä ja voi osittain käyttää samoja REST-verbejä kuin HTTP, kuten GET, POST, PUT ja DELETE. Tämän yhtenäisyyden vuoksi CoAP:n ja HTTP:n integroiminen toisiinsa on helppoa.

REST-verbeistä lisää luvussa 3.4.5. CoAP voidaan salata sisäänrakennetulla DTLS-salauksella, mutta suurin ongelma CoAP:n käyttämisessä IoT-protokollana ei ole tietoturva, vaan kuljetuskerros UDP:n epävarma tiedonsiirto. [3, s. 186−187.]

3.4.3 MQTT

MQTT on kevytrakenteinen tuottaja-tilaajamallin kommunikointiprotokolla, joka kulkee TCP-yhteyden päällä. MQTT suunniteltiin alun perin toimivaksi myös epävakaissa verkoissa, joissa esiintyy korkeita vasteaikoja ja pieniä kaistanleveyksiä ja sen suunniteltu käyttökohde oli kerätä telemetriaa ja sensoriarvoja öljyputkistoista satelliittiyhteyksien avulla SCADA-järjestelmään melkein 20 vuotta sitten. MQTT-nimi on aiemmin tullut sanoista Message Queue Telemetry Transport, mutta uusimman protokollaa ylläpitävän standardin mukaan lyhenteen tarkoituksesta on luovuttu ja protokollan nimi on yksinkertaisesti MQTT. [20, s. 31; 22.]

Tiedonsiirto eri päätepisteiden välillä MQTT:n avulla on aihepohjaista. Tuottaja- ja tilaajajäsenten välinen tiedonsiirto toteutetaan käyttämällä kolmatta, välittäjäjäsentä. Välittäjä muodostaa aiheen, johon tuottaja työntää dataa pyytämättä. Tilaajajäsen voi tilata aiheen tai osan siitä, jolloin vain haluttu data siirtyy tilaajalle. MQTT:n perusversiossa ei ole sisäänrakennettua salausta, mutta tiedonsiirto voidaan salata kuljetuskerroksella TCP-yhteyden TLS-/SSL-salauksella ja päätepisteiden välille voidaan myös muodostaa VPN-tunneli (Virtual Private Network). MQTT on kevyen rakenteensa, tuottaja-tilaaja-mallinsa ja yksinkertaisuutensa vuoksi ideaalinen valinta IoT-protokollaksi. [3, s. 187.]

3.4.4 AMQP

Advanced Message Queueing Protocol (AMQP) on suunniteltu alun perin finanssi- ja pankkialan protokollaksi, joka toimii sovellusten ja/tai organisaatioiden välillä. Pankkien tietoliikenteen protokollana AMQP:ssä on panostettu erityisesti tietoturvaan ja luotettavaan tiedonsiirtoon. Protokollan liikenne voidaan suojata SASL- ja/tai TLS-salauksella. AMQP on MQTT:n tapainen tuottaja-tilaaja-mallin protokolla ja aihepohjainen viestien jako toimii pitkälti samalla tavalla, mutta AMQP perustuu myös nimensä mukaisesti viestijonoihin. Viestijonoarkkitehtuurin avulla protokollan viestit tulevat varmasti perille ja aikajärjestyksessä. AMQP:n käyttö soveltuu erityisen hyvin silloin, kun tietoliikenteessä ilmenee valtava määrä pienikokoisia viestejä, joiden siirron toteutuminen on

äärimmäisen tärkeää, joten protokolla on hyvä valinta datakriittisiin teollisen internetin ratkaisuihin. [3, s. 188; 23.]

3.4.5 HTTP(S)

HTTP (Hypertext Transfer Protocol) ja sen salattu versio HTTPS (HTTP Secure) ovat nykyisten verkkoselainten tiedonsiirron perusta. Protokolla toimii kyselymallin mukaisesti. Kun jokin verkkosivu avataan verkkoselaimella, haetaan HTTP:n avulla eri verkkopalvelimilta kaikki ne kuvat, videot, skriptit, dokumentit, yms. joista verkkosivu koostuu, ja selainohjelmisto esittää nämä yhtenäisenä sivuna.

HTTP:n käyttö tiedonsiirtoprotokollana IoT-sovelluksissa ja -laitteissa perustuu REST-arkkitehtuurin (Representational State Transfer) mukaiseen rajapintaan, joka löytyy melkein kaikista IoT:n ja teollisen internetin sovelluksista. REST-arkkitehtuuri muodostaa API:n (Application Programming Interface) eli ohjelmointirajapinnan käyttäen URI-osoitetta (Uniform Resource Identifier) ja REST-verbejä. REST-API on IoT-palveluntarjoajan luoma ja uniikki kullekin sovellukselle, joten sen ominaisuudet, kuten URI-osoitteen muodostus, erilaiset parametrit ja tarvittavat otsakkeet, tietoturva ja tuetut REST-verbit ovat palveluntarjoajan käsissä. [20, s. 29−30.]

IoT-sovelluksissa yleisesti tuettuja REST-verbejä ovat GET, POST, PUT ja DELETE. Roy Fielding määritteli RESTin osana tohtorintutkintoaan vuonna 2000 ja oli pääkirjoittaja HTTP/1.1 versiossa, joka on edelleen laajassa käytössä. Standardissa RFC2616, joka koskee HTTP/1.1 versiota, on määritelty GET-, POST-, PUT- ja DELETE- verbien geneerinen toiminta:

- GET-verbillä asiakas pyytää palvelinta lähettämään URI-osoitteen määrittelemää informaatiota. GET-kyselyn on tarkoitus vain palauttaa dataa, eikä sillä saisi olla muita toimintoja.
- POST-verbillä asiakas pyytää palvelinta vastaanottamaan URI-osoitteen määrittelemää informaatiota ja kirjoittamaan sen URI-osoitteen määrittelemän resurssiin. Informaatio sisällytetään POST-kyselyyn.
- PUT-verbillä asiakas pyytää palvelinta vastaanottamaan informaatiota ja luomaan uuden URI-osoitteen määrittelemän resurssin sekä kirjoittamaan PUT-kyselyn sisältämän informaation tähän resurssiin. Mikäli resurssi on jo olemassa, informaatio kirjoitetaan olemassa olevan resurssin päälle.
- DELETE-verbillä asiakas pyytää palvelinta poistamaan URI-osoitteen määrittelemän resurssin. [24; 25, s. 53−56.]

Nämä ovat siis standardin määritelmiä, mutta toteutus on täysin IoT-palveluntarjoajasta kiinni. Esimerkkinä voidaan ottaa ThingWorx-pilvipalvelun tapa käyttää POST-verbiä. ThingWorxissä POST-verbillä suoritetaan alustalle kirjoitettuja ohjelmanpätkiä POST-kyselyn sisältämien parametrien mukaan ja näin mahdollistetaan esimerkiksi usean muuttujan kirjoittaminen yhden HTTP-viestin avulla. [26.]

HTTP-viestin muodostamisessa käytetään myös otsakkeita, niin kutsuttuja headereitä. Otsakkeissa voidaan esimerkiksi määrittää HTTP-viestin sisältöä POST- ja PUT-kyselyille, sisällön tietotyyppi Content-Type-otsakkeella tai viestin autentikointi Authorization-otsakkeella. [24, s. 31.] GET-kyselyn muotoisen HTTP-viestin muodostamista GE Historian -ohjelmiston REST-rajapintaan ja tämän viestin rakennetta on kuvattu esimerkkikoodissa 1.

```
GET  /historian-rest-api/v1/datapoints/currentvalue?tagNames=tagName1 HTTP/1.1
Host:     https://<historianservername>:8443
Headers:  Accept: application/json
          Authorization: Bearer <token>
```

Esimerkkikoodi 1. Esimerkki HTTP-viestin muodostamisesta GE Historian -ohjelmiston REST-rajapinnalle. <historianservername> täytetään vastaamaan palvelimen verkko-osoitetta ja <token> täytetään OAuth2-tokenilla. [27, s. 37−38.]

HTTP RESTin ja muiden asiakas-palvelinmallin mukaisten protokollien käyttäminen IoT-protokollana on osittain ongelmallista tuottaja-tilaajamalliin verrattuna. Tuottaja-tilaajamallissa data liikkuu ennalta määrätyn aikavälin tai arvomuutoksen seurauksena automaattisesti, mutta REST-rajapintaa käytettäessä kyselyt pitää erikseen automatisoida ja konfiguroida juoksemaan esimerkiksi tietyin aikavälein. Tämä voi aiheuttaa turhaa tiedonsiirtoa, mikäli kyselyn pyytämä data ei ole muuttunut aikavälin aikana. Nämä ongelmat ovat kuitenkin lähinnä REST-rajapinnan toteutuksesta riippuvaisia. Omat ongelmansa REST-rajapinnan käyttämiseen tuovat myös erilaiset viestien autentikointiin käytettävien tokenien elinajat, kuten OAuth2-autentikoinnin tapauksessa.

3.4.6 OPC UA

OPC UA:n käyttö tiedonsiirtoon teollisen internetin sovelluksissa on luonnollinen valinta, sillä se mahdollistaa liittymisen suoraan teolliseen prosessiin ja takaa erittäin hyvän tietoturvan. Tiedonsiirto OPC-palvelimien ja -asiakkaiden välillä voidaan hoitaa nopealla TCP:n päällä kulkevalla binäärisellä protokollalla tai Web Service -pohjaisella SOAP-HTTPS-protokollalla. OPC UA:n käyttöä ainoana tietoliikenneprotokollana teollisen

internetin sovelluksissa rajoittaa sitä tukevien pilvialustojen vähäinen määrä. Microsoft julkisti vuonna 2016 ottavansa OPC UA:n käyttöön omissa ympäristöissään, myös Azure-pilvipalvelussaan. [19; 28.]

OPC UA:n käyttäminen teollisen internetin sovelluksissa onkin tällä hetkellä suuntautunut enemmän datan aggregointiin ja tietoturvalliseen linkkiin automaatioverkon ja DMZ-alueen välille OPC UA -tunneloinnilla [9]. OPC-säätiön OPC UA -työryhmä julkisti lisäävänsä standardiin tuottaja-tilaajamallin mukaisen kommunikaation, joka toteutetaan AMQP- tai MQTT-protokollalla. Lisäys on vastaus teollisen internetin nopeaan kasvuun sekä Teollisuus 4.0 -mallin mukaisten tehtaiden tiedonsiirtoon. Tämä lisäys vahvistaa OPC UA:n asemaa merkittävästi teollisen internetin tiedonsiirtoratkaisuna. [29.]

3.5 Siirrettävän datan muoto ja malli

Sensorien tuottama ja mahdollisten ohjelmistojen yhteen kokoama IoT-data siirretään pilvipalveluille tietoliikenneprotokollien avulla, mutta protokollat eivät määrittele kantamansa sisällön muotoa. Data on siirrettävä sellaisessa muodossa, jota pilvialusta tukee ja tämä on pääkriteeri muodon valinnassa. Muutamia yleisiä tietomuotoja tähän tarkoitukseen ovat CSV, XML, JSON ja palveluntarjoajien omat, binääriset tietomuodot.

CSV (Comma Separated Value) ei ole standardoitu tiedostomuoto, mutta on laajasti käytössä yksinkertaisen rakenteensa vuoksi erilaisen taulukkomuotoisen sisällön tallentamisessa. Nimensä mukaisesti CSV-tiedoston sisältämä data on eroteltu pilkuin. Yksi taulukkorivi muodostuu esimerkiksi merkki- tai numerojonoista, jonka tietuet on eroteltu pilkuilla. Useat taulukkorivit ovat eroteltu rivinvaihdolla. CSV ei ole paras mahdollinen tietotyyppi IoT-datan siirtämiseen, sillä se ei helposti tue hierarkkista dataa. [30.]

XML (Extensible Markup Language) on merkintäkieli, joka määrittelee ihmis- ja koneluettavan tiedostomuodon. XML on erittäin kattava ja tukee hierarkkista dataa, mutta se on myös raskas tiedostomuoto. Tämän vuoksi XML ei välttämättä ole oikea ratkaisu pienten, prosessointikapasiteetiltaan rajoitettujen IoT-laitteiden tiedostomuodoksi. [30.]

JSON (JavaScript Object Notation) on avoimen standardin tiedostomuoto. Vaikka nimi tuleekin JavaScript-ohjelmointikielestä, on se kielestä riippumaton, kompakti ja kevyt tiedostomuoto. JSONista on muodostunut de facto -standardi verkkosovelluksien ja myös

IoT:n sekä teollisen internetin tiedonsiirtoon. JSON-muodossa dataa voidaan esittää muuttujien avulla, jossa tietylle muuttujanimelle sidotaan erilaisia tietotyyppejä numeroista merkkijonoihin ja taulukkorakenteisiin. [30.]

Binäärisellä tiedostomuodolla tarkoitetaan tiedostomuotoa, jonka sisältö ei välttämättä ole tekstinä, vaan binäärisessä muodossa ja täten ei ole ihmisen luettavissa. IoT-palveluntarjoajien tai -laitevalmistajien käyttämät binääriset tiedostomuodot ja myös protokollat tarkoittavat käytännössä suljettuja rajapintoja, jotka ovat haitaksi sovelluskehitykselle, mutta nopeuttavat tiedonsiirtoa huomattavasti. [30; 31.]

Siirrettävän datan muodon lisäksi pitää huomioida tietomalli. Automaatiolaitteilta luettava data noudattaa yleensä mallia, jossa ilmaistaan muuttujan nimi ja sen hetkellinen arvo, sekä UTC-aikaleima millisekunteina (tai jokin muu aikaleima), jolloin muuttujan arvo luettiin tai se viimeksi muuttui. OPC-standardin mukaisesti malliin lisätään myös Quality-, eli laatuarvo. Laatu ilmaistaan Good- tai Bad-arvona, käytännössä boolina, joka kuvaa OPC-palvelimen yhteyttä automaatiolaitteeseen. [32, s. 49.] Esimerkkikoodissa 2 on ilmaistu tyypillinen OPC-palvelimen lähettämä tietomalli JSON-muodossa, muokattuna luettavampaan muotoon.

```
{
      "Data":[{
            "timestamp":1521795404605,
            "values":[
                  {
                        "id":"Simulator01.Device01.TagName01",
                        "v":999,
                        "q":true,
                        "t":1521795404605
                  },
                  {
                        "id":"Simulator01.Device01.TagName02",
                        "v":94,
                        "q":true,
                        "t":1521795404605
                  }
            ]
      }]
}
```

Esimerkkikoodi 2. OPC-palvelimen JSON-tietomuoto. Data-taulukossa on viestin lähetysajankohdan aikaleima , sekä values-taulukko, joka sisältää kaksi tietuetta. Tietueina ovat kahden eri muuttujan, TagName01 ja TagName02 ID-, Value-, Quality- ja Timestamp-arvot. Muuttujien aikaleimat ovat mitattuja aikaleimoja ja voivat erota hierarkiassa ylempänä löytyvästä koko viestin aikaleimasta. Muuttujien hierarkkinen rakenne ilmaistaan ID:ssä pistein eroteltuna.

Datan mallintaminen on yksi suurimmista haasteista teollisen internetin sovelluksissa. Kuvaavien muuttujanimien määrittely automaatiojärjestelmässä on erittäin tärkeää, koska niiden perusteella dataa kirjoitetaan myös historiatietokantaan ja pilveen. Pitkien muuttujanimien käyttäminen ei välttämättä silti ole aina hyödyksi, sillä jokainen väliaskel, kuten historiatietokanta tai OPC-palvelin, lisää omat määrittelynsä muuttujan nimeen. Muuttujien nimeäminen jonkin hierarkkisen mallin mukaan on käytännöllistä, kun ajatellaan niiden käyttämistä ylemmillä tasoilla. ISA-95-standardin soveltaminen nimeämiskäytäntöön on järkevää, jolloin muuttujat erotellaan kohteen, alueen, linjaston, työsolun ja laitteen perusteella. Tämä on käytännöllinen lähestymistapa, kun dataa aletaan analysoimaan. Tiedetään arvon alkuperä ja voidaan esimerkiksi etsiä korrelaatioita linjaston sisällä. Jokaista muuttujaa ei tietenkään nimetä kattorakenteelta asti, vaan näiden nimeäminen voidaan hoitaa edellä mainituilla OPC-palvelimilla ja tietokannoilla, joissa erilaiset ryhmä- ja muut hierarkkiset rakenteet ovat helposti toteutettavissa. [9; 10, s. 147.]

IoT:n tietomallina käytetään yleisesti niin sanottua digitaalista kaksosta. Tämä tarkoittaa yksinkertaistettuna laitteen arvojen, parametrien, mittojen yms. digitaalista esittämistä pilvessä. Otetaan kuluttajapuolen IoT-laitteesta esimerkki: älykello. Älykello saattaa kerätä tietoa GPS-sijainnista; paine-, kosteus- ja kiihtyvyys antureista; bluetooth- ja WLAN-yhteyksistä sekä monista muista lähteistä, mukaan lukien käyttäjäsyötteet. Vaikka älykellovalmistaja valmistaisi useampaa mallia älykellostaan, voivat ne perustua samoihin anturi- ja käyttöliittymäratkaisuihin. Data-analyysin helpottamiseksi voidaan dataa kerätä ennalta määritellyn tietomallin mukaan ja jaotellaan se tietokantaan kaksosmallin mukaisesti. Analyysin lisäksi ohjelmointi helpottuu olennaisesti, sillä siirrettävä data on valmiiksi strukturoitua. Digitaalisen kaksosen yhtenä mahdollisena ominaisuutena on myös ohjelmapätkien suorittaminen IoT-laitteella, kuten etäpäivitysten suorittaminen. [9.]

3.6 Paikallisten historiatietokantojen hyödyntäminen

Historiatietokannat ovat aikasarjadataa keräämiä tietokantoja, joita on yleisesti käytössä myös automaatioalan ulkopuolella. Historiatietokanta yhdistetään eri ohjelmistoihin ja järjestelmiin, jonka jälkeen tietokantaan kerätään haluttujen muuttujien arvoja. Automaatiossa nämä ovat yleisesti mittaussignaaleja, PLC-ohjelmien muuttujia sekä SCADA- ja MES-järjestelmien arvoja, kuten eränumeroita ja muita tunnisteita. [9.]

Moni teollisuuden yritys kerää jo automaatioverkostaan sensori- ja muuttuja-arvoja keskitettyyn, mutta paikalliseen historiatietokantaan. Tämän historiatietokannan hyödyntäminen teollisen internetin sovelluksissa käy järkeen, jos tarkoituksena on siirtää prosessin arvoja pilveen analyysiä varten. Historiatietokannat mahdollistavat myös vanhan, ennen IoT-sovelluksen luontia syntyneen datan siirtämisen pilveen analyysin tueksi ja kohteeksi. Teollisen internetin sovelluksissa niiden käyttäminen tuo tiedonsiirtoon mukaan enemmän viivettä, kuin suoraan OPC-palvelimelta siirtäminen. Voidaan pohtia myös molempien tekniikoiden yhdistelmän käyttöä, jossa aikakriittisimmät arvot viedään suoraan pilveen ja paikalliselta historiatietokannalta kysellään haluttuja arvoja esimerkiksi raporttien luomista varten, tai ratkaisua jossa historiatietokanta sijaitsee pilvessä. Ratkaisu, jossa hyödynnetään pilvipohjaista historiatietokantaa, on tämän insinöörityön pääaihe. [9.]

4 IoT:n ja teollisen internetin pilvialustat

Aiemmin luvussa 3.1 kerrottiin teollisen internetin teknologiapinon alustatasosta ja pilvipalveluista. Tämä luku avaa käsitettä käytännöllisemmästä ja teollisen internetin näkökulmasta.

4.1 Pilvipalvelut yleisesti

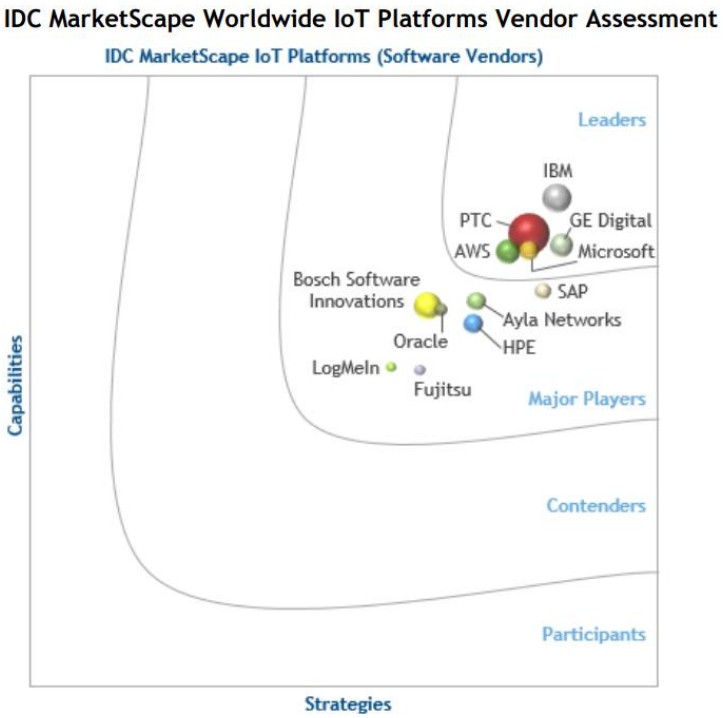
Pilvipalvelut nykyisessä muodossaan saivat alkunsa vuonna 2005, kun yhdysvaltalainen verkkokauppajättiläinen Amazon julkaisi AWS-palvelunsa (Amazon Web Services). Tarkoituksena oli myydä konesalien ja infrastruktuurin ylijäämää, kuten laskentatehoa ja tallennustilaa yrityksen ulkopuolisille tahoille. Liiketoimintamallina oli laskentatehon ja tallennustilan myyminen niiden käyttöasteen mukaan, ja malli on edelleen käytössä Amazonin lisäksi monella muulla pilvipalveluita toimittavalla yrityksellä. Liiketoimintamalli oli ja on edelleen erityisesti pienten ja keskisuurten yritysten toimintoihin sopiva, sillä niiden ei tarvinnut investoida suuria summia omiin datakeskuksiin ja konesaleihin. Innovaatiot ja kehitys pilvipalveluiden saralla on yksi tärkeistä IoT:n mahdollistajista. [21, s. 47−49.]

Pilvipalvelut voidaan yleisellä tasolla jakaa kolmeen palveluluokkaan, joista pilvipalveluita toimittavat yritykset voivat toimittaa yhtä tai useampaa. IaaS-luokan (Infrastructure as a Service) palvelut tarjoavat nimensä mukaisesti infrastruktuuria palveluna. Asiakas ostaa suoraa laskentatehoa, tallennustilaa tai verkkoinfrastruktuuria. Nämä palvelut ovat yksinkertaisimmat, joita palveluntarjoaja tarjoaa. Esimerkiksi Microsoftin Azure-pilvipalvelussa asiakas voi luoda Windows- tai Linux-virtuaalikoneen ja maksaa tästä kuukausittaista maksua sen päälläoloajan, tallennustilan käytön ja verkkoliikennemäärän mukaan. PaaS-luokan (Platform as a Service) palvelut tarjoavat itse pilvialustaa palveluna. PaaS-palveluita tarvitaan silloin, kun rakennetaan pilvialustan päälle jotain muuta sovellusta tai ratkaisua. Palveluntarjoajien analytiikkatyökalut, jotka sijaitsevat pilvessä ovat myös PaaS-luokan palveluita. SaaS-luokan (Software as a Service) palvelut tarjoavat pilvessä sijaitsevaa ohjelmistoa palveluna. Hyvin yksinkertainen esimerkki SaaS-luokan palvelusta ovat internetin eri sähköpostipalvelut. Laajan teollisen internetin tai IoT-sovelluksen luominen käyttää käytännössä kaikkia pilvipalveluluokkia. Infrastruktuuria tarvitaan datan varastointiin, alustan avulla toteutetaan data-analyysi ja tuotetaan pilvessä toimiva ohjelmisto asiakkaita varten. [21, s. 47−49.]

4.2 Sovellukseen sopivan pilvialustan valinta

Teollisen internetin sovellusta suunniteltaessa ja rakentaessa pitää melko aikaisessa vaiheessa päättää, mitä pilvialustaa käytetään. Sovelluskehityksen näkökulmasta alustavalinnan kriteereitä voivat olla esimerkiksi alustan analytiikan ja sovelluskehityksen työkalut, rajapinnat alustalle niin sisään- kuin ulospäin ja alustan tietoturvallisuus. Liiketoiminnan kannalta kriteerejä voivat olla kaupallisen alustan kustannukset tai avoimen lähdekoodin perustuvien alustojen kehitystyön kustannukset. Myös yrityksen muut järjestelmät voivat asettaa kriteerejä. Mikäli yritys käyttää liiketoimintansa tukena jotakin pilvipalvelua, olisi tietysti luontevaa, että myös suunniteltava sovellus rakennettaisiin kyseisen alustan päälle tai sellaiselle, joka tukee yhteyttä alustojen välille. [3, s. 233−235.]

Kaupallisten alustojen markkinaosuudet ja tulevaisuuden näkymät voivat myös olla merkittävä valintakriteeri. Koska pilvipalvelut ovat nimensä mukaisesti palveluita, niiden ylläpitämien yritysten taloudellinen menestys voi olla ratkaisevassa asemassa alustan jatkokehityksen, ylläpidon ja tuen näkökulmista. International Data Corporationin (IDC) vuonna 2017 tekemän IoT-alustojen tutkimuksen (kuva 6) mukaan markkinoiden johtavissa asemissa ovat IBM:n Watson, PTC:n ThingWorx, GE Digitalin Predix, Microsoftin Azure ja Amazon Web Services. [33.]



Kuva 6. IDC:n tekemä markkinoilla olevien IoT-alustojen tutkimus vuodelta 2017 [33].

Insinöörityössä tutustuttiin kahteen Novotek Oy:n käyttämistä pilvialustoista: PTC:n ThingWorx ja Microsoftin Azure. Työn rajaamiseksi pääpaino pidettiin Azure-alustassa.

4.3 PTC ThingWorx

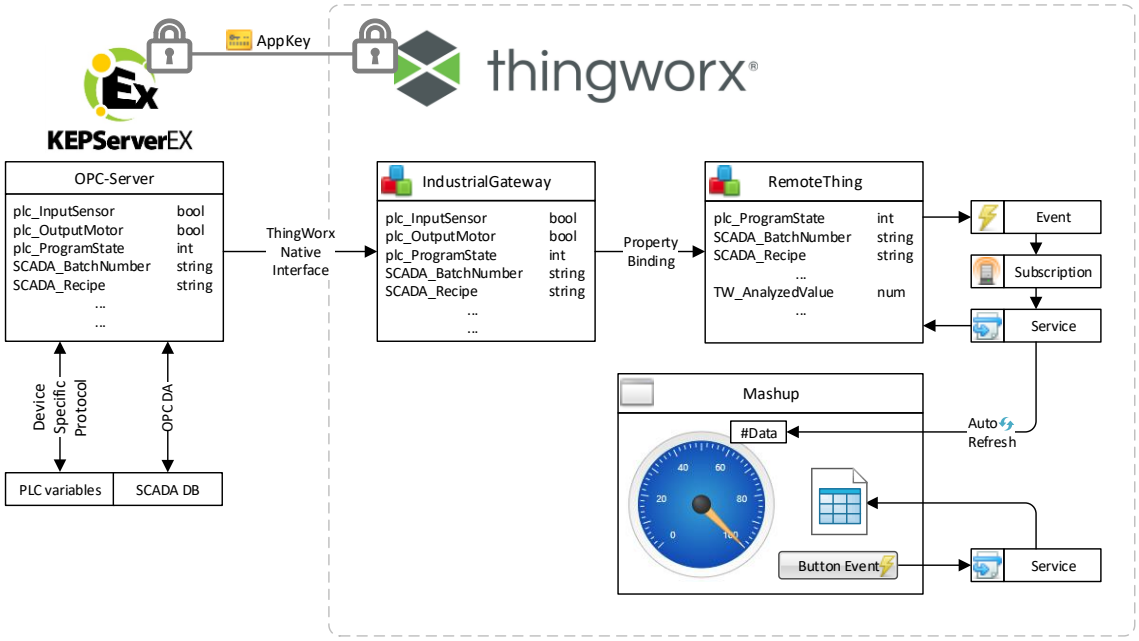
ThingWorx on yksi vanhimmista, kaupallisista, puhtaasti IoT:hen keskittyvistä pilvialustoista. ThingWorxin ensimmäinen versio julkaistiin vuonna 2013 ja yhdysvaltalaisperäinen PTC osti sen saman vuoden lopussa. Alusta mahdollistaa helpon ja nopean yhdistämisen teolliseen prosessiin. ThingWorx tarjoaa PaaS-luokan pilvipalveluita data-analytiikasta AR-sovelluksiin (Augmented Reality, lisätty todellisuus). PTC myy ThingWorx-alustaa esimerkiksi Amazonista ja Microsoftista poiketen ohjelmistona. Asiakas asentaa ohjelmiston omaan konesaliinsa tai ostaa jonkin muun pilvipalveluntarjoajan IaaS-palveluita. Suomessa Novotek myy ThingWorx-alustaa PTC:n jäälleenmyyjänä. Myös Elisa myy ThingWorx-alustaa ja -palvelua Elisa IoT -nimikkeellä [34; 35.]

4.3.1 ThingWorx-perusteet

Kuten pilvipalveluilla on tapana, myös ThingWorxin käyttöliittymänä toimii verkkoselain. ThingWorxin ydinidean mukaan koko alusta toimii olio-ohjelmoinnista tutun objektimallin mukaan. ThingWorxin perusolio on nimeltään Thing. Thing perustuu aina johonkin Thing Template -malliin ja voi myös täyttää erillisiä Thing Shape -malleja. Nämä mallit voidaan rinnastaa olio-ohjelmoinnin luokkarakenteeseen ja eri perintömalleihin. Jokaisella Thingillä on muuttujia ja metodeja, joiden avulla alustalle luodaan sisältöä JavaScript-ohjelmointikielellä. Alustalla voidaan myös luoda helposti näyttöjä verkkosivujen muodossa Mashup-olioilla. Näytöille tuotetaan sisältöä käyttämällä Mashup Builder -työkalua, jonka Widget-kirjastosta löytyy valmiita palikoita esimerkiksi yksittäisen arvon näyttämiseen mittareilla tai trendin näyttämiseen Graph-työkalulla. Kirjastosta löytyy monia valmiita työkaluja ja niiden lisääminen on helppoa ThingWorx Marketplacen kautta tai omia työkaluja kirjoittamalla ThingWorxin Software Development Kiteillä (SDK) C-, Java-, .NET-, iOS- ja Android-ohjelmointiin. Lisäosien kirjoittaminen ja lisääminen ei toki ole vain Mashuppien ominaisuus vaan koko alusta on laajennettavissa. [34; 36.]

4.3.2 ThingWorx-yhteys

ThingWorx mahdollistaa yhteyden periaatteessa millä vain protokollalla aiemmin mainittujen lisäosien avulla. Sisäänrakennettuja protokollia alustalla on HTTP(S) REST ja PTC:n kehittämä suljettu AlwaysON-protokolla. Erityisesti valmistavan teollisuuden näkökulmasta tärkein yhteyden muodostus -tapa hoidetaan erillisellä PTC:n KEPServerEX-ohjelmistolla. KEPServerEX on teollisuuden johtava OPC- ja yhteysohjelmisto, joka mahdollistaa suoran yhteyden ThingWorx-alustaan käyttäen AlwaysON-protokollaa ThingWorx Native Interface -ajurin kautta. KEPServerEX tukee yli 150 teollista laitekohtaista protokollaa ja rajapintaa, joten sillä pystytään liittymään käytännössä mihin tahansa teolliseen laitteeseen. Yhteyden muodostamiseen tarvitaan ThingWorx-alustalla luotu autentikointiavain sekä päätepiste, johon KEPServerEX alustalla liittyy. Tämä päätepiste on IndustrialGateway Thing Template -mallin mukaan luotu Thing, jonka Discovery- ja Property Binding -toiminnoilla voidaan teollisen prosessin muuttujat sitoa ThingWorx-alustan Thingien muuttujiin. Näiden muuttujien arvojen perusteella ja Thingien metodeilla voidaan sitten luoda Mashup-näyttöjen sisältöä. [32; 36, s. 24.] Tätä yleiskuvaa on pyritty havainnollistamaan paljolti yksinkertaistetussa kuvassa 6.



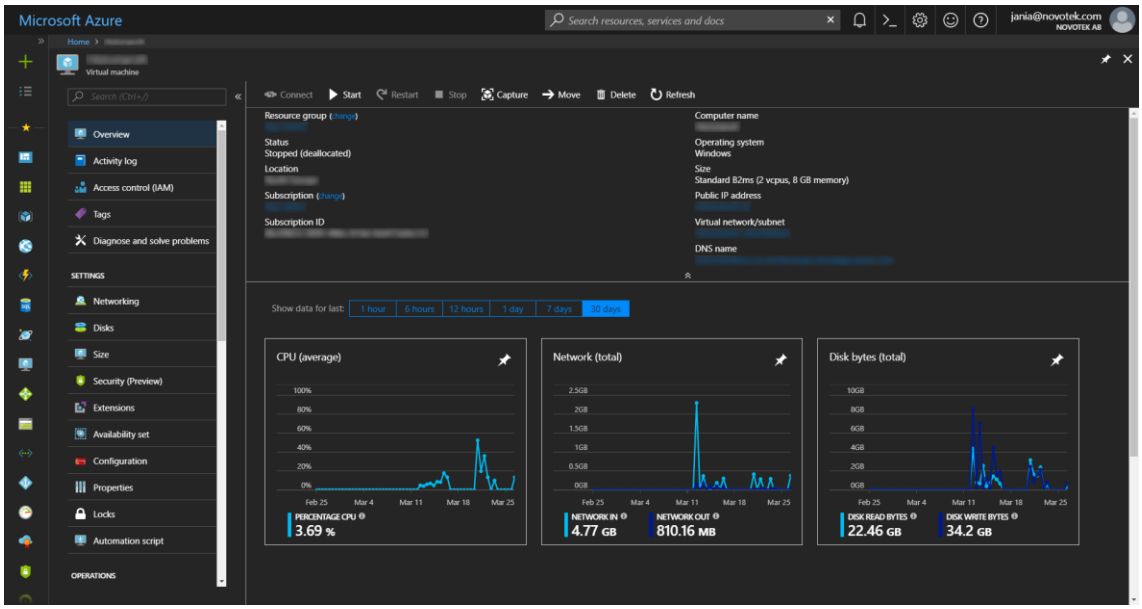
Kuva 7. Tietoliikenne automaatiolaitteilta KEPServerEXin avulla ThingWorx-alustalle. Alustalla JavaScript-ohjelmapätkien ja tapahtumien avulla tieto analysoidaan ja siirretään Mashup-näytöille.

4.4 Microsoft Azure

Microsoft Azure on yhdysvaltalaisen ohjelmistojättiläisen Microsoftin pilvipalvelu. Azure julkaistiin vuonna 2010 nimellä Windows Azure, mutta nimettiin Microsoft Azureksi vuonna 2014. IoT-sovellukset mahdollistava Azure IoT Hub julkaistiin vuoden 2016 alussa. Azuressa on yli 600 erilaista palvelua, joista näkyvimmät liittyvät dataan, sen varastointiin ja analysoimiseen. Tallennus-, analyysi- ja hallintapalveluita sekä -työkaluja löytyy monelle eri tietokannalle sekä SQL- että NoSQL-tyypin tietokannoille. Azuren palveluiden hinnoittelu perustuu yksinkertaistettuna palveluiden käyttöasteeseen, prosessointikapasiteettiin sekä tietovarastojen kohdalla myös redundanttiseen tallennuksen tarpeeseen. [37.]

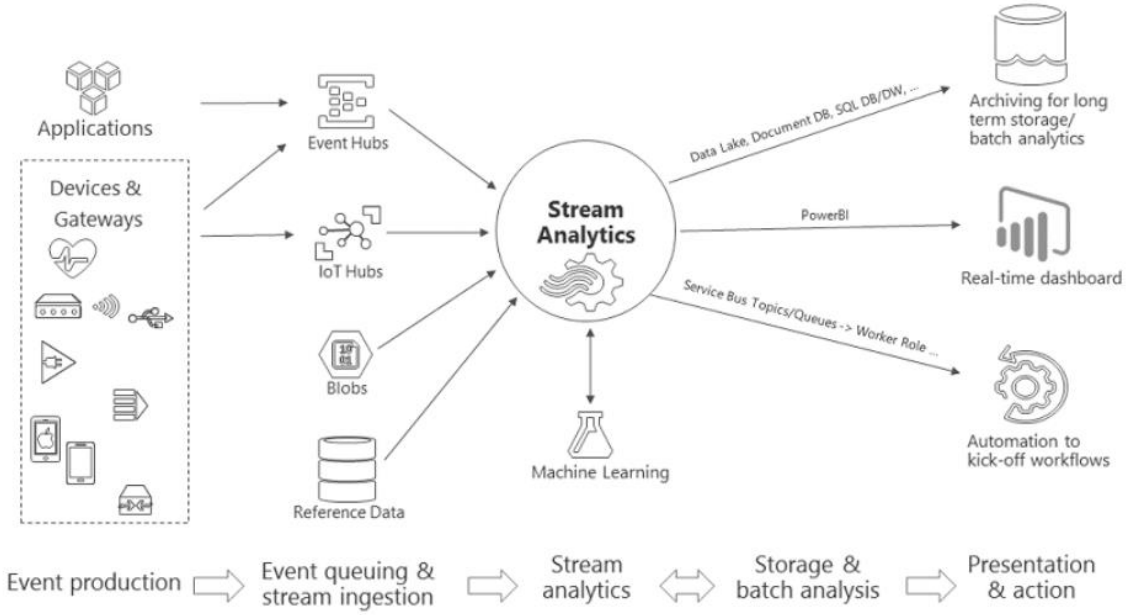
4.4.1 Azure yleisesti

Azuren peruskäyttöliittymänä toimii verkkoselain (kuva 8). Käyttöliittymästä voidaan luoda infrastruktuuripalveluita, kuten uusi virtuaalikone tai tietoliikenneputki kahden tai useamman muun palvelun välillä. Suuri osa Azuren toiminnallisuuksista voidaan myös hoitaa komentorivin kautta tai Azuren omalla REST APIlla. Azure tarjoaa myös alustaansa sovelluksenkehittäjille ja IoT:n näkökulmasta tarjonta on hyvin data- ja analyysipainotteista. Alusta mahdollistaa myös liittymisen moneen muuhun Microsoftin palveluun ja ohjelmistoon, kuten Power BI -tiedonkäsittelyohjelmistoon. [37.]



Kuva 8. Azure-käyttöliittymä virtuaalikoneen ylläpitoon.

Alustan IoT-päätepisteenä toimii Azure IoT Hub. IoT Hubiin liitytään AMQP-, MQTT- tai HTTPS-protokollilla, joko suoraan laitteelta, pilvessä sijaitsevan protokollamuuntimen tai paikallisen IoT Edge -laitteen avulla. IoT Hub toimii eräänlaisena datan aggregoijana ja mahdollistaa digitaalisten kaksosten käytön. Microsoft hinnoittelee IoT Hubin alustalle päivittäin lähetettävien viestien lukumäärän mukaan, joten suuresta määrästä dataa pieninä viesteinä voi tulla ongelma alustaa käytettäessä. IoT Hubista data siirretään Azure Stream Analytics -työkalun avulla Azuren muihin työkaluihin ja/tai tietovarastoihin. Stream Analyticsin avulla voidaan myös siirrettävälle datalle tehdä nopeaa analyysiä ennen tietovarastoon tallentamista tai visualisointia. Stream Analytics -työkalua on havainnollistettu kuvassa 9. [38.]



Kuva 9. Azure Stream Analytics -työkalun mahdolliset informaatiovirrat [38].

4.4.2 Azure IoT Hub

Azure-yhteyksien luontiin käytetään Microsoftin Device Explorer -työkalua, joka on erillinen ohjelma Azuren IoT Hub -laitteiden luontiin. Työkalulla yhdistytään Azureen luotuun IoT Hubiin sen verkko-osoitteella ja yhteysavaimella. Uudelle IoT-laitteelle annetaan työkalussa nimi sekä luodaan SAS-token (Shared Access Signature), jota käytetään tiedonsiirron autentikointiin. SAS-tokenille määritetään elinaika, jonka suurin mahdollinen arvo on 365 päivää tietoturvallisuussyistä. [39.]

Kuten ThingWorx-alustan kanssa, myös Azureen on helppo liittyä teollisesta prosessista PTC:n KEPServerEX-ohjelmistolla. Ohjelmistosta löytyy IoT Gateway -lisäosa, joka mahdollistaa MQTT- ja HTTPS-REST-asiakkaiden sekä HTTPS-REST-palvelimen konfiguroinnin. Microsoft suosittelee MQTT:n käyttöä RESTin sijasta IoT Hubiin liityttäessä. IoT Gatewayn MQTT-asiakas konfiguroidaan liittymään Azureen sen IoT Hub -verkko-osoitteella ja Device Explorerissa luoduilla IoT-laitteen nimellä sekä SAS-tokenilla. Konfiguroinnissa voidaan myös vaikuttaa muun muassa viestien lähetystaajuuteen sekä MQTT:n Quality of Service -asetukseen, joka määrittelee, lähetetäänkö paketit maksimissaan kerran varmistamatta niiden perillepääsyä; vähintään kerran varmistuksen kera vai tasan kerran varmistuksen sekä kuittauksen kera. Myös muuttumattomien arvojen lähetys on konfiguroitavissa, jolloin voidaan vähentää turhaa tiedonsiirtoa merkittävästi. [40.]

4.4.3 Azure Stream Analytics

Stream Analytics on tapahtumien prosessointityökalu, johon on mahdollista syöttää jatkuvasti dataa. Työkalun konfiguroinnissa määritellään Stream Analyticsille yksi tai useampi syöte ja yksi tai useampi lähtö. Syötteinä työkalu tukee Azuren Event Hubia, IoT Hubia sekä Blob Storagea. Event Hub on samantapainen työkalu kuin IoT Hub ja Blob Storage on Azuren sisäinen tiedostontallennuspalvelu, jonne voidaan tallentaa minkä vain muotoista dataa. Lähtöjä Stream Analyticsistä löytyy muun muassa Event Hub, erilaiset tietovarastot kuten SQL- ja NoSQL-tietokannat, Blob Storage- ja Data Lake -varastot sekä suoraan tietoliikennevirtaa hyödyntävät Power BI ja Time Series Insights -työkalut. [41.]

Azuren verkkokäyttöliittymässä Stream Analytics -työkaluun konfiguroidaan halutut inputit ja outputit, jonka jälkeen muodostetaan kysely, jolla IoT-data voidaan suodattaa ja ohjata haluttuihin analytiikan työkaluihin. Kysely muodostetaan Microsoftin kehittämällä T-SQL-kyselykielellä, joka on nimensä mukaisesti SQL-kieleen perustuva kyselykieli. Yksinkertaisin kysely siirtää kaikki viestit syötteestä lähtöön (esimerkkikoodi 3). [41.]

```
SELECT *
INTO [Output]
FROM [Input]
```

Esimerkkikoodi 3. Yksinkertainen Stream Analytics -kysely, joka siirtää kaiken datan syötteestä lähtöön ilman suodatusta.

4.4.4 Azuren tietovarastot

Azure mahdollistaa monen eri tietovaraston ja tietokannan käytön. Microsoft tukee monen eri SQL-variantin käyttöä sekä myös NoSQL-tietokantoja, kuten Cosmos DB ja MongoDB. Pilvi-infrastruktuurin käyttäminen mahdollistaa myös virtuaalikoneiden pystyttämisen ja täten lähes minkä tahansa tietokannan ja tietovaraston käyttö on mahdollista, vaikka Microsoft ei olisi siitä suoraa työkalua tehnytkään. Azure mahdollistaa myös datan kopioinnin tietokannoista ja -varastoista toiseen muun muassa Data Factory -palvelun avulla. Myös tiedonvarastointi erilaisiin tauluihin ja tiedostoihin on mahdollista Azure Storage -palveluita käyttämällä. Yksi näistä palveluista on Blob Storage, joka on tarkoitettu tiedostojen tallentamista varten kansiomalliseen rakenteeseen. Tiedostoja voivat olla mitä vain tekstitiedostoista videoihin ja IoT-dataan. IoT:n näkökulmasta tallennettavat tiedostot ovat lokitiedostoja, joihin kirjoitetaan IoT-viestien sisältö. Esimerkkikoodin 3 mukaisen Stream Analytics -kyselyn Blob Storageen kirjoitettava tiedosto on ajan mittaan kasvava JSON-tiedosto. [42.]

4.4.5 Azure koneoppiminen

Koneoppiminen on viime vuosina esille noussut analytiikkatyökalu ja tekoälyn osa-alue. Koneoppimisella tarkoitetaan algoritmeja, jotka nimensä mukaisesti opettavat konetta toimimaan paremmin ja halutulla tavalla. Teollisen datan analysoinnissa voidaan käyttää tekoälyä esimerkiksi ennakoivan huollon sovelluksiin. Azure-pilvipalvelu mahdollistaa omien koneoppimissovellusten luonnin Machine Learning Studio -palvelulla. Tämä tukee syötteinä esimerkiksi Azuressa sijaitsevia ja paikallisia SQL-tietokantoja sekä muita tietokantoja ja lähteitä. [43.]

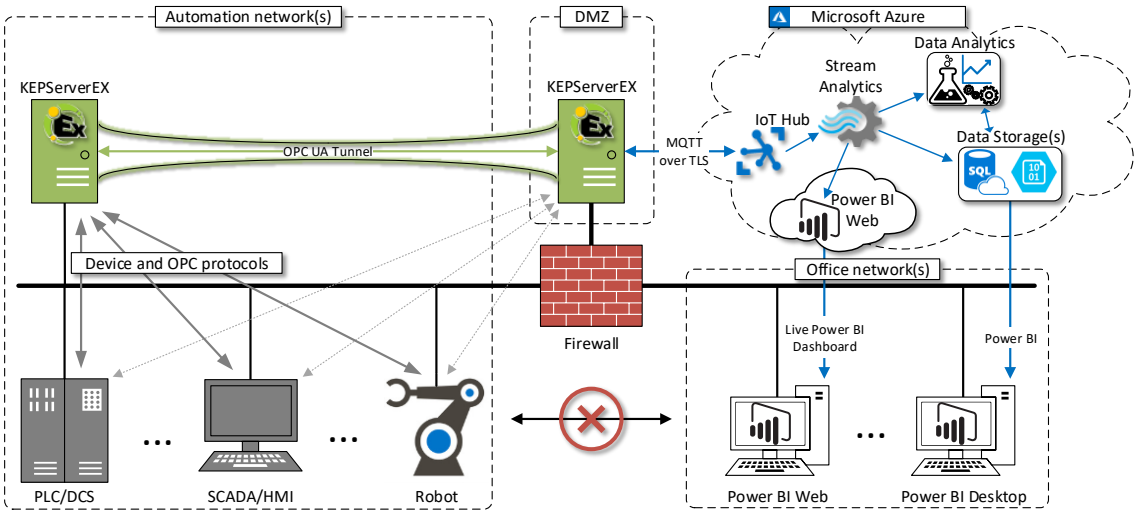
5 Novotekin ratkaisut

Insinöörityötä aloitettaessa tutustuttiin Novotekin ratkaisuihin, joilla teollista dataa siirretään pilveen ja mahdollistetaan sen analysoiminen ja raporttien tekeminen. Työn alkumetreillä ei ollut vielä päätetty, mitä pilvialustaa työssä käytettäisiin, mutta Novotekille kohdistuvan kasvavan Azure-kysynnän vuoksi oli luonnollista ja ajankohtaista keskittyä Azureen. Työn alussa tutustuttiin kuitenkin myös ThingWorx-alustaan.

Novotek on PTC:n jälleenmyyjä Suomessa, joten on luonnollista, että yrityksen teollisen datan tiedonsiirtoratkaisut perustuvat PTC:n markkinajohtavaan OPC-ohjelmistoon nimeltä KEPServerEX. Yritys toimittaa myös General Electricin ohjelmistotuotteita, joista yksi on GE Historian. Historianin Remote Collector -toiminnot sekä uusimman 7.0-ohjelmistoversion tuoma REST-rajapinta mahdollistavat myös Historianin käytön teollisen datan tiedonsiirrossa internetin yli.

5.1 KEPServerEX – Azure IoT Hub

KEPServerEX-ohjelmistoon pohjautuvaa mallia, jossa KEPServerEX hoitaa tiedonsiirron yrityksen DMZ-alueelta pilvipalveluun, on havainnollistettu kuvassa 10.



Kuva 10. KEPServerEX-pohjainen tiedonsiirtomalli teolliselle datalle Azure-pilvipalveluun.

Asiakkaan automaatioverkkoon lisätään palvelin, jolle asennetaan KEPServerEX-ohjelmisto, jonka ajureilla sekä rajapinnoilla liitytään automaatiojärjestelmän eri osiin. Toinen

KEPServerEX asennetaan DMZ-alueelle ja nämä kaksi instanssia yhdistetään toisiinsa OPC UA -tunnelilla. Mikäli automaatioverkkoon ei ole mahdollista lisätä palvelinta, voidaan myös DMZ-alueelta liittyä automaatioverkkoon laitekohtaisilla ajureilla, rajapinnoilla ja protokollilla. Tämä ei ole yhtä tietoturvallinen vaihtoehto kuin OPC UA -tunnelointia hyödyntävä kahden KEPServerEX-instanssin malli. [9.]

Azure-pilvialustalle luodaan IoT Hub -päätepiste, joka konfiguroidaan KEPServerEXin IoT Gateway -lisäosalle ja tiedonsiirto hoidetaan MQTT-protokollalla TLS-salauksella. Myös VPN-yhteyden muodostaminen Azureen on mahdollista. Stream Analytics -työkalulla päätetään datan päätepisteet ja tallennetaan Azuren omiin tietovarastoihin. Power BI -ohjelmistolla voidaan lukea dataa Azuren tietovarastoista sekä luoda raportteja teollisesta datasta. [9.]

Sopivan tietovaraston valinta Azuressa on asiakkaan päätös, mutta käytännöllisimmältä ja järkevimmiltä vaikuttavat Azure Data Lake -palvelut. KEPServerEXin lähettämä data on niin sanottua litteää (flat) dataa, eli muodossa ei ole hierarkiaa. Rakennetta lisätään ohjelmiston päässä konfiguroimalla halutut automaatiojärjestelmän muuttujat omiin ryhmiin, joka lisää ryhmän nimen muuttujanimen eteen. Tätä rakennetta puretaan Stream Analytics -työkalulla haluttuun muotoon ja suodatetaan valittuihin tietovarastoihin. [9.]

IoT-datan, joka on esimerkkikoodin 2 (sivu 19) mukaisessa muodossa, suodattamiseen tarkoitetun T-SQL-kyselyn muodostamisessa pitää ottaa myös huomioon datan JSON-muoto. Mikäli data halutaan tallentaa esimerkiksi SQL-tietokantaan, Data-taulukko muodostaisi oman kolumninsa, sillä normaalisti taulukkoa käsiteltäisiin yhtenä merkkijonona. Tämä ei ole ihanteellista, joten JSON-rakenne pitää purkaa Stream Analytics -kyselyllä sopivampaan muotoon. SQL-tietokantaan muodostetaan ensiksi taulu, joka määritellään yhdeksi Stream Analyticsin lähteistä. Tauluun luodaan tarvittavat kolumnit ja nimetään ne. T-SQL-kyselyllä JSON-rakenne puretaan viittaamalla array-taulukkoon, jonka sisällä ne ovat. Suodattaminen suoritetaan SQL-kielestä tutulla WHERE-lauseella. Rakenteen purkaminen ja datan jakaminen eri lähtöihin voidaan hoitaa esimerkkikoodin 4 mukaisella kyselyllä.

```
SELECT
    [arrayvalues].ArrayValue.t AS [timestamp],
    [arrayvalues].ArrayValue.id AS [id],
    [arrayvalues].ArrayValue.v AS [value],
    [arrayvalues].ArrayValue.q AS [quality]
INTO [PowerBIOutput]
FROM [KEPInput] AS [kep]
CROSS APPLY GetArrayElements([kep].[values]) AS [arrayvalues]
WHERE [values].ArrayValue.id = 'Simulator01.Device01.TagName01'


SELECT
    [arrayvalues].ArrayValue.t AS [timestamp],
    [arrayvalues].ArrayValue.id AS [id],
    [arrayvalues].ArrayValue.v AS [value],
    [arrayvalues].ArrayValue.q AS [quality]
INTO [SQLOutput]
FROM [KEPInput] AS [kep]
CROSS APPLY GetArrayElements([kep].[values]) AS [arrayvalues]
WHERE [values].ArrayValue.id = 'Simulator01.Device01.TagName02'
```

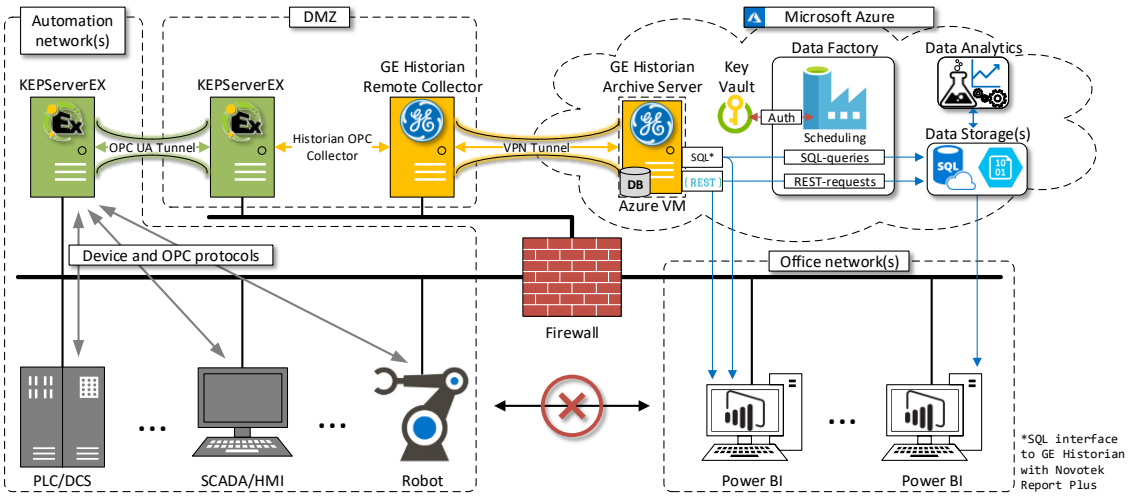
Esimerkkikoodi 4. Stream Analytics -kysely, jolla KEPServerEXin JSON-muotoinen viesti puretaan ja jaetaan kahteen eri lähtöön, SQL-tietokantaan ja Power BI:hin. Komennot korostettu sinisellä ja merkkijonot oranssilla.

5.2 KEPServerEX – Historian – Azure

Teollisuudessa käytetään usein jotain historiatietokantaa tallentamaan historiallista dataa teollisesta prosessista. Tätä dataa on hyödynnetty tehtaan sisäisesti eräraportointiin, kunnossapitoon ja moneen muuhun tarkoitukseen jo vuosia ennen IoT:n tuloa. [9.]

Novotek toimittaa GE Historian -ohjelmistoa, joka on markkinajohtava historiatietokanta. Tietokanta pystyy keräämään aikasarjadataa monesta eri lähteestä SCADA-järjestelmistä eri OPC-spesifikaatioihin ja sitä voidaan käyttää jopa palvelinten suorituskyvyn ja tallennustilan seuraamiseen. Keräily onnistuu myös etänä Remote Collectorien avulla. [45.] Etäkeräily mahdollistaa myös tiedonsiirtomallin, jossa Historian-palvelin on asennettu pilvessä pyörivään virtuaalikoneeseen ja etäkeräilijäohjelmisto kerää dataa teollisesta prosessista. Yksi tärkeimmistä mallin tuomista eduista on se, että etäkeräilijäohjelmiston ja pilvipohjaisen Historianin välisen yhteyden ei tarvitse olla aina auki tai edes vakaa. Mikäli yhteys katkeaa tai on epävakaa teknillisistä tai muista syistä, Collector-ohjelmisto tallentaa keräämänsä datan omaan, väliaikaiseen tietokantaansa, kunnes yhteys avautuu uudelleen. Kuten KEPServerEX, myös Historian tukee osittain ja täysin redundanttisia asennuksia, jolloin palvelinrikko ei merkitse tietoliikenteen ja datan tallentamisen katkeamista. Toinen etu Historianin käyttöön tiedonsiirrossa on sen uusimmasta 7.0 versiosta löytyvä REST-rajapinta, jonka avulla voidaan lukea tietokannan sisältämää dataa HTTP-viesteillä. [9.]

Historiania käyttävää tiedonsiirtomallia on avattu kuvassa 11. Kuvassa on eroteltu KEPServerEX- ja Historianin etäkeräilijä -ohjelmistot erillisille palvelimille DMZ-alueella selvyyden vuoksi, mutta ohjelmistot voidaan kuitenkin asentaa samalle palvelimelle. Historian-palvelimen ei myöskään tarvitse sijaita pilvessä, vaan Data Factoryn REST-kyselyt voidaan suorittaa internetin yli. Tämä mahdollistaa esimerkiksi palvelun myymisen asiakkaille, joilla on jo Historian asennettuna tehtaalla ja haluavat siirtää historiadatan Azuren pilvivarastoihin.



Kuva 11. Historian-ohjelmistolla toteutettu tiedonsiirtomalli.

5.3 Mallien erot

KEPServerEX-malliin pohjautuva tiedonsiirto on yksinkertaisempi. Yhdistyksiä tarvitsee tehdä vain yksi ja konfigurointien määrä rajoittuu vain KEPServerEX-ohjelmistojen väliseen OPC UA -linkkiin ja muuttujien hierarkkisen rakenteen tekemiseen. Tiedonsiirto ei kuitenkaan ole aukotonta. Mikäli yhteys Azureen katkeaa, saattaa dataa jäädä siirtämättä siitä huolimatta, että KEPServerEXistä löytyy Store and Forward -toiminto datan väliaikaiseen tallentamiseen. Microsoftin hinnoittelupolitiikka on myös monimutkainen ja IoT Hub sekä Stream Analytics saattavat osoittautua erittäinkin kalliiksi vaihtoehdoksi suurilla tietomäärillä. GE Historian -mallin tiedonsiirto on monimutkaisempi, mutta historiatietokanta on luotettava tietovarasto. Myös tiedonsiirto etäkeräilijän ja palvelimen välillä on taattu, dataa ei hukata vaikka yhteys katkeaisikin. Historianiin on myös sisäänrakennettuja datan tiivistystoimintoja ja käyttäjienhallinta. Molempien mallien yhdistäminen kokonaisuudeksi lienee paras vaihtoehto.

6 GE Historian ja Azure Data Factory

Novotek pyrkii lisäämään Historian-ohjelmiston käyttöä IoT-ratkaisuissaan tietovaraston asemassa. ja tätä tiedonsiirtomallia olikin jo testattu ja käytetty ennen insinöörityön aloittamista. Ongelmana Historian-mallissa ei ollut tiedonsiirto kentältä pilveen, vaan pilvensisäisesti. Työssä toteutettiin Historian Remote Collector -pohjainen tiedonsiirto kentältä Azureen sekä Historianin REST-rajapintaa hyödyntävä tiedonsiirto Azuren sisäisesti Azure Data Factory -työkalulla.

6.1 Testausinfrastruktuuri

Azure-pilvipalveluun pystytettiin uusi virtuaalikone, jolle asennettiin Historian-ohjelmisto. Virtuaalikoneen palomuuri ja Azuren virtuaaliverkot konfiguroitiin päästämään liikenne läpi tarvittavista porteista. Novotekin paikalliselle palvelimelle luotiin virtuaalikone, jolle asennettiin KEPServerEX- ja Historian OPC Collector -ohjelmistot. KEPServerEXille konfiguroitiin testauksia varten muutama simuloitu muuttuja, joita luettiin Historian OPC Collectorilla OPC DA -yhteyden yli. Collector konfiguroitiin liittymään Azure-pilvipalvelussa sijaitsevaan Historianiin tälle annetun DNS-nimen avulla. KEPServerEX-ohjelmistoon lisätyt simulaattorimuuttujat konfiguroitiin päivittämään arvonsa kerran sekunnissa. Historian Collector lukee muuttujat samalla, yhden sekunnin resoluutiolla. Data saatiin vaivattomasti liikkeelle ja sitä pystyttiin tarkastelemaan etätyöpöytäyhteyden avulla Azuren virtuaalikoneelta.

6.2 REST-rajapinnan rakenne ja viestien autentikointi

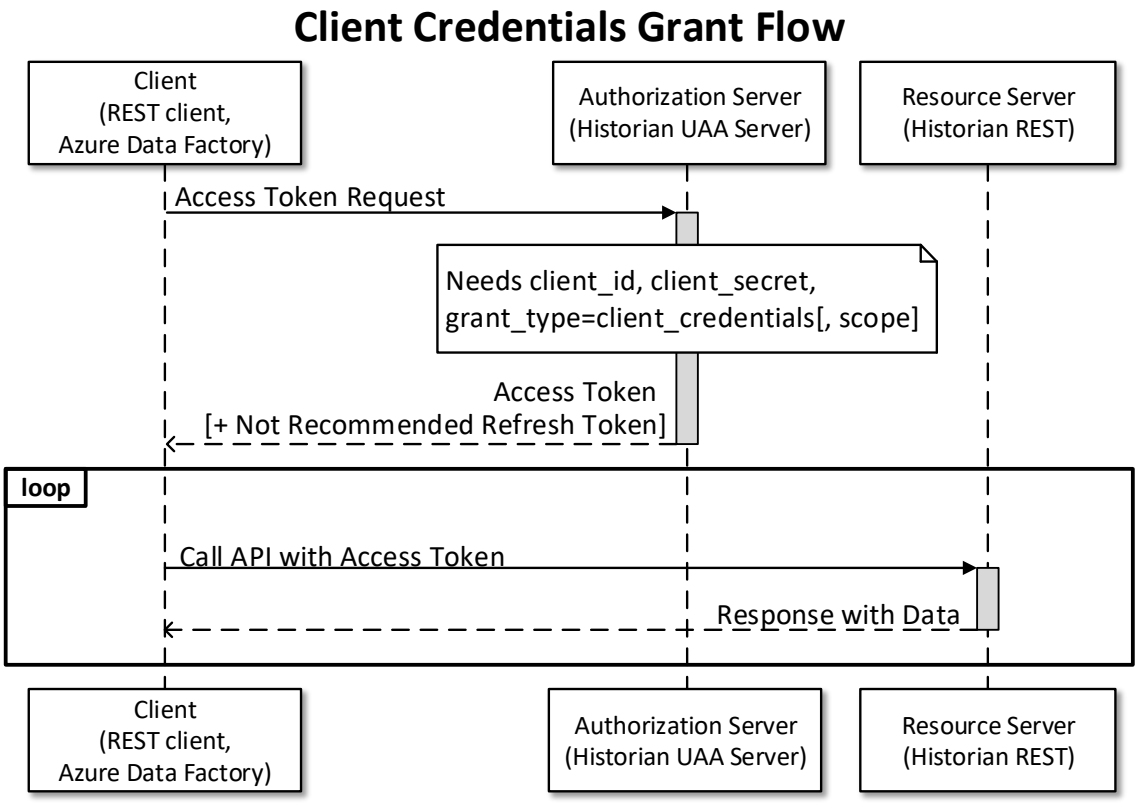
GE Historianin REST API on rakennettu osittain hyödyntäen Cloud Foundry -nimisen avoimen lähdekoodin pilvipalvelun tekniikkaa, jota GE käyttää myös omassa Predix-pilvialustassaan. Tärkein hyödynnetty osa on Cloud Foundryn käyttäjien ja viestien autentikointiin tarkoitettu UAA-palvelin (User Account and Authentication Service). Yksinkertaisuudessaan UAA rakentuu näkyvyysalueista ja käyttäjistä. Näkyvyysalueisiin määritetään käyttöoikeuksia ja käyttäjille annetaan oikeudet kyseisiin näkyvyysalueisiin. UAA:n näkyvyysalueita ovat esimerkiksi UAA-palvelimen toimintoja sisältävä uaa-alue ja asiakkaiden clients-alue. Historianin tapauksessa uniikkeja näkyvyysalueita ovat REST-

rajapinnan luku- ja kirjoitusoikeudet sisältävä historian_rest_api- ja Historianin oman verkkokäyttöliittymän historian_visualization-alueet. [27, s. 5−6; 45, s. 100−103.]

Historianin REST-viestien autentikointi tapahtuu OAuth2-viitekehyksellä. OAuth2 on avoin standardi, jolla kolmannelle osapuolelle annetaan oikeus verkkopalveluun tai sen sisältöön ilman, että käyttäjätunnuksia ja salasanoja tarvitsee antaa kolmannen osapuolen käyttöön. OAuth2 on käytössä monella tunnetulla verkkopalvelulla, kuten Facebookilla, GitHubilla ja Googlella. RFC 6749 -standardi, joka ylläpitää OAuth2-viitekehystä, määrittelee neljä roolia, jotka osallistuvat autentikointiin

- Resource Owner, resurssin omistaja omistaa autentikoinnin kohteena olevan resurssin ja sen sisällön
- Resource Server, resurssin palvelin, jolta kolmas osapuoli haluaa sisältöä
- Client, asiakas on kolmas osapuoli, yleensä jokin sovellus tai verkkosivu, joka pyytää autentikointia
- Authorization Server, autentikointipalvelin antaa oikeudet eli OAuth2-tokenin asiakkaalle, jolla asiakas voi pyytää resurssin palvelimelta haluamansa sisällön. [46, s. 6; 47.]

Historianin tapauksessa resurssin palvelin on sen REST-palvelin, asiakas on sovellus, joka kysyy REST-viesteillään sisältöä, ja autentikointipalvelimena toimii UAA-palvelin [27, s. 5−6]. Historian käyttää RFC 6749 -standardin määrittelemää Client Credentials -autentikointia, jossa asiakas on käytännössä resurssin omistaja [46, s. 9]. Autentikointi on kuvattu kuvan 12 vuokaaviossa. Asiakas, joka tämän insinöörityön näkökulmasta on Azuren Data Factory -työkalu, pyytää OAuth2-tokenin Historianin UAA-palvelimelta käyttäjätunnuksella ja salasanalla, jonka jälkeen dataa noudetaan Historianin REST-palvelimelta käyttäen OAuth2-tokenia autentikointiin.



Kuva 12. OAuth2-viitekehyksen mukainen autentikointi Historianin REST-rajapinnalle [47, muokattu].

OAuth2-tokeneita on kahta mallia, Access ja Refresh, joista Historian käyttää RFC 6749 -standardin mukaisesti vain Access-tokeneita Client Credentials -autentikoinnissaan. Tokenilla on oletuksena 12 tunnin elinaika, jonka jälkeen se pitää luoda uudestaan UAA-palvelimelta. Mikäli token luodaan uudestaan ennen elinajan päättymistä, vanha token mitätöidään. Token luodaan UAA-palvelimelle lähetettävällä REST-kyselyllä, joka vaatii kuvan 12 mukaisesti client_id:n ja client_secretin, jotka ovat asiakkaan käyttäjänimi ja salasana. Oletuksena vain Historianin admin-pääkäyttäjällä on oikeudet REST-rajapintaan, ja näiden tunnusten käyttäminen tiedonsiirtoon ei ole tietoturvallinen ratkaisu. Käyttäjänimestä ja salasanasta muodostetaan HTTP-viestin Basic-autentikointiin 64-bittisellä binaarimuunnoksella salattu merkkijono, mutta tämä muunnos on käännettävissä, eikä vaadi moderneilta tietokoneilta montaakaan millisekuntia. [47, s. 10−12; 48.]

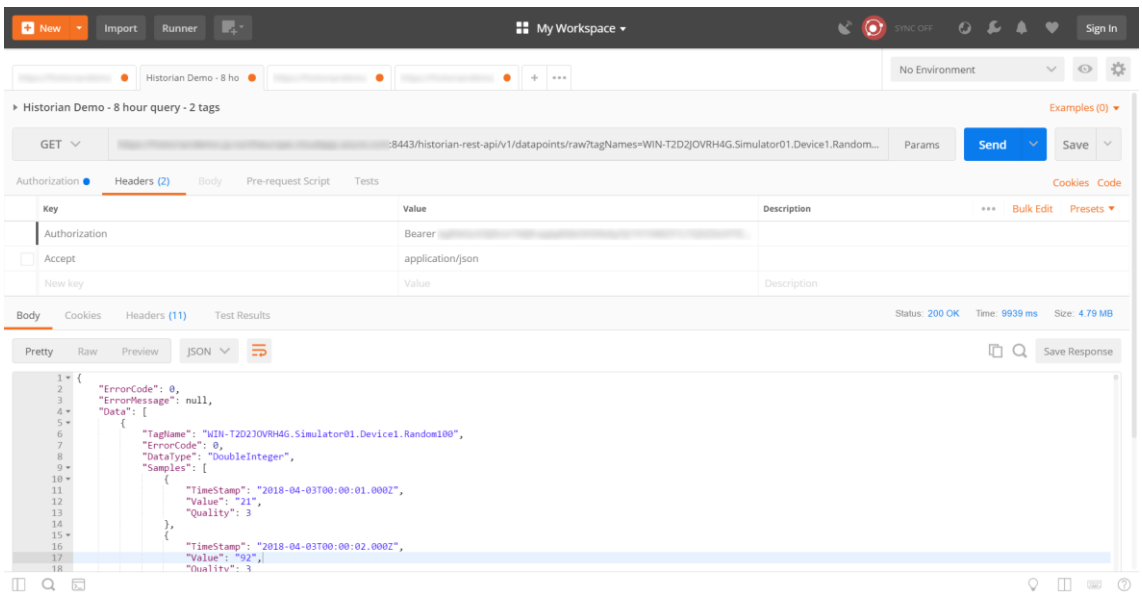
REST-rajapinnan tietoturvallisempaa käyttöä varten Historianin UAA-palvelimelle lisättiin uusi asiakas Ruby-komentorivityökalulla ja asiakkaalle annettiin oikeudet vain historian_rest_api.read-näkyvyysalueeseen. Näin parannettiin tietoturvaa tilanteessa, jossa luodun asiakkaan käyttäjätunnus ja salasana vuotaisivat ulkopuoliselle taholle. Salasana

tallennettiin Azure Key Vault -palveluun. Asiakasta lisättäessä olisi voitu myös vaihtaa OAuth2-tokenin elinaikaa, mutta myös tämän katsottiin olevan tietoturvariski Data Factory -työkalun puutteiden vuoksi. Mikäli token vuotaisi, voisi ulkopuolinen taho lukea Historianin REST-rajapinnalta dataa. Tätä riskiä voidaan minimoida palomuuriasetuksilla.

6.3 REST-kyselyt

Historianin REST-rajapinta tukee GET-, POST-, PUT- ja DELETE-verbejä, joista tärkein on GET. GET-verbillä voidaan lukea minkä tahansa Historian-muuttujan arvot halutulla aikavälillä. Kyselyt voidaan parametrisoida monella eri tapaa, haluttu data voidaan esimerkiksi hakea ns. raakana tai interpoloituna datana. Voidaan myös päättää, haetaanko vain viimeiset n-määrä arvoja tai vain pienin tai suurin arvo joltain aikaväliltä. Muuttujilta voidaan myös hakea metatietoa. Kaikki Historianin REST-vastaukset ovat JSON-muodossa. [26, s. 7.] Tyypillinen haku (esimerkkikoodi 1) on kuitenkin jonkin tietyn muuttujan tai tiettyjen muuttujien arvot halutulla aikavälillä.

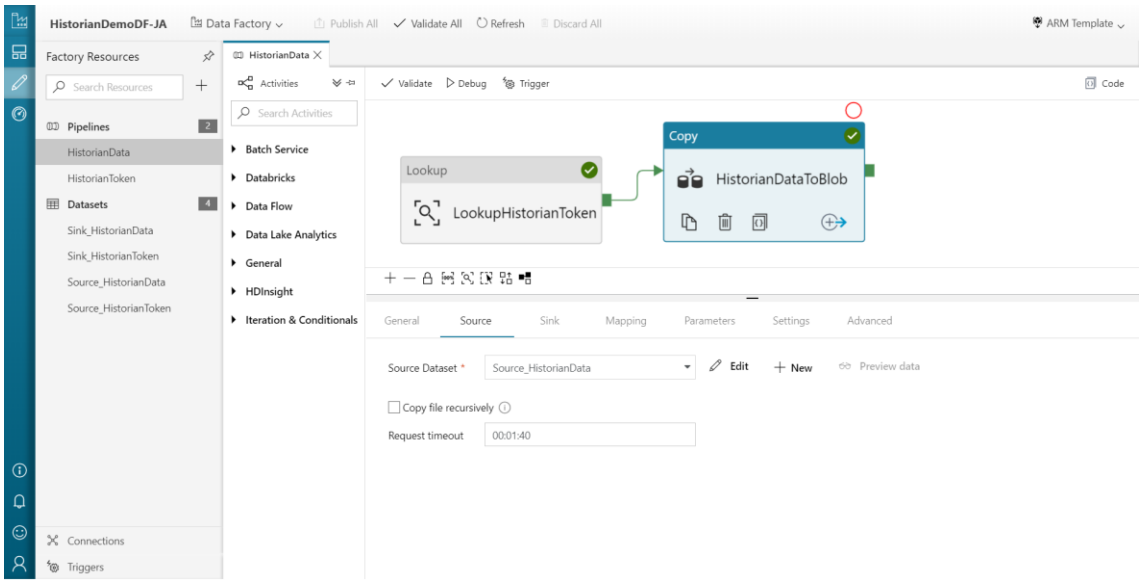
Historianin REST-rajapinnan tarkasteluun ja testaamiseen käytettiin Postman-työkalua, jolla voitiin helposti muodostaa REST-kyselyitä graafisesta käyttöliittymästä (kuva 13). Postman on sovelluskehitykseen tarkoitettu työkalu, jolla voidaan lähettää REST-kyselyitä halutulle palvelimelle sekä näyttää vastausten sisältöä.



Kuva 13. Postman-työkalun käyttöliittymä. REST-kysely, jolla noudetaan kahden muuttujan tiedot 8 tunnin ajalta.

6.4 Azure Data Factory

Insinöörityön tarkoituksena oli mahdollistaa tietoliikenne Azure-pilvipalvelun sisäisesti Historianin REST-rajapinnalla. Tähän tarkoitukseen löydettiin Azure Data Factory -työkalu (kuva 14) ja luotiin siitä oma instanssi Azureen. Data Factory on tarkoitettu datan kopiointiin yhdestä tietokannasta toiseen. Tietokantojen ei tarvitse olla samassa pilvipalvelussa tai ollenkaan pilvessä eikä edes saman tyyppisiä, eli data voidaan lukea esimerkiksi NoSQL-tyypin tietokannasta ja tallentaa SQL-tyypin tietokantaan. Datan lukeminen on mahdollista noin 70:stä eri lähteestä ja tallentaminen noin 20 paikkaan. Kopiointi tapahtuu Pipeline-kokonaisuuksissa, jotka koostuvat eri aktiviteeteista, syötteistä ja lähdöistä. Melkein kaikki parametrit Data Factoryssa voidaan muodostaa dynaamisesti, esimerkiksi aikaleiman muodostaminen REST-kyselyyn tai tiedon tallentaminen automaattisesti numeroituun ja kasvavaan tiedostoon. [49.]



Kuva 14. Azure Data Factoryn verkkokäyttöliittymä.

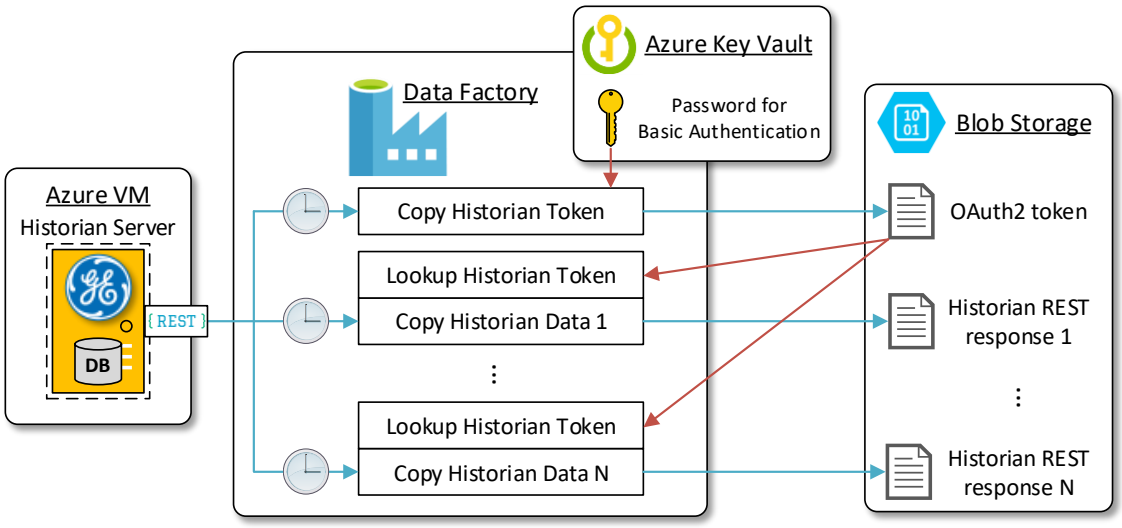
Data Factoryyn luotiin uusi Copy Data -aktiviteetti, jonka lähteeksi valittiin HTTP, eli REST-rajapinta. Lähde konfiguroitiin liittymään Historianin REST-rajapintaan Basic-autentikoinnilla aiemmin luodulla, uuden UAA-asiakkaan tunnuksilla. Autentikointia varten tarvittava salasana luettiin automaattisesti Azure Key Vault -työkalusta joka kerta, kun kysely muodostetaan uudestaan. Näin Data Factory ei ikinä itse tiedä salasanaa, vaan Key Vault antaa sen pyydettäessä. Autentikoitu kysely saa REST-vastauksessa OAuth2-tokenin JSON-muodossa, jonka tallennuspaikaksi valittiin Azure Blob Storage. Blob Storage ei ole kaikkein tietoturvallisin paikka tokenin tallentamiseen, mutta oikeilla

käyttäjäoikeuksien rajauksilla, sekä Historian-koneen palomuuriasetuksilla tietoturvaa voidaan parantaa. Token kirjoitettiin aina samaan tiedostoon Blob Storagessa, jolloin siihen viittaaminen muissa Pipeline-toiminnoissa oli helpompaa. Pipeline-toiminnolle konfiguroitiin 12 tunnin ajastin, jolloin token luodaan ja haetaan uudestaan, kun se oletuselinajan jälkeen erääntyy.

Data Factoryyn luotiin toinen Copy Data -aktiviteetti, jolla siirrettiin itse dataa. Tämänkin lähteeksi valittiin HTTP ja tallennuspaikaksi Blob Storage. Uuteen Pipeline-toimintoon lisättiin Lookup-aktiviteetti, jolla luettiin Blob Storagesta juuri luotu OAuth2-token. Tokenilla muodostettiin dynaaminen REST-kysely, joka pyytää Historianilta kaiken datan viimeisen 8 tunnin ajalta kahdelle muuttujalle. Kyselyä ja sen dynaamista muodostamista on kuvattu esimerkkikoodissa 5. Valmis tiedonsiirtorakenne kuvassa 15.

```
GET   /historian-rest-api/v1/datapoints/raw?tagNames=WIN-
T2D2JOVRH4G.Simulator01.Device1.Random100;WIN-
T2D2JOVRH4G.Simulator01.Device1.Sin01&start=@{addhours(utcnow(),-
8)}&end=@{utcnow()}&count=0&direction=0    HTTP/1.1
Host:       https://<historianservername>:8443
Headers:    Accept:          application/json
            Authorization:   Bearer @{activity('LookupHistorianToken').
                             output.firstRow.access_token}
```

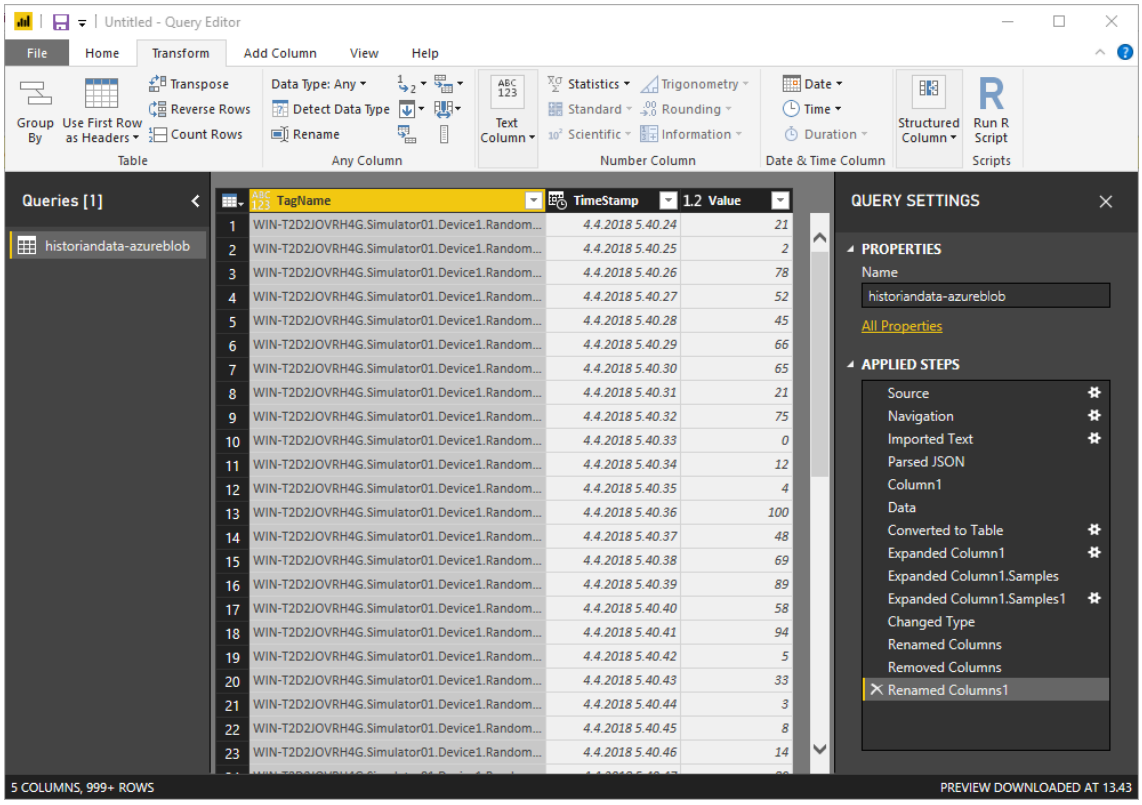
Esimerkkikoodi 5. REST-kyselyn muodostaminen Azure Data Factoryssa käyttäen sinisellä korostettuja dynaamista sisältöä ja funktioita. Authorization-otsakkeen koodirivillä noudetaan Lookup-aktiviteetissa määritelty Blob Storagessa sijaitsevasta JSON-tiedostosta OAuth2-token.



Kuva 15. Tiedonsiirtokaavio Historianin REST-rajapinnasta Blob Storageen.

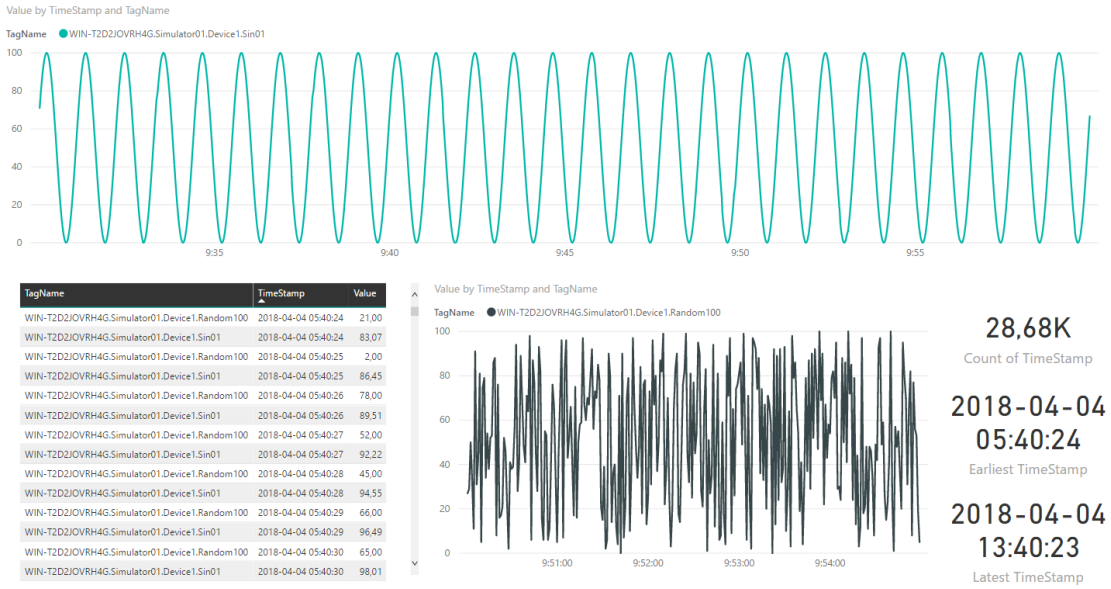
6.5 Power BI

Kun data oli saatu liikkeelle ensin paikalliselta virtuaalikoneelta ja lopuksi Azuren sisäisesti Data Factory -työkalulla, siirryttiin insinöörityön viimeiseen vaiheeseen eli Datan visualisointiin. Visualisoinnissa käytettiin Microsoftin Power BI -ohjelmistoa, joka mahdollistaa yksityiskohtaisten kyselyiden muodostamisen moneen eri tietokantaan ja tietorakenteeseen. Power BI on Microsoftin yrityksille suuntaama datan analysointi- ja tiedonkäsittelyohjelmisto. Ohjelmisto tukee satoja tietolähteitä, myös monia Azuressa sijaitsevia lähteitä. Power BI:llä voi analysoida melkein minkä vain muotoista dataa ja luoda tästä raportteja yrityksen käyttöön. Yleinen käyttökohde on yritysten bisnesosastot, jotka analysoivat yrityksen liiketoimintaa. Myös teollinen data voidaan analysoida Power BI:llä ja luoda esimerkiksi tuotannon eräraportteja. [9.] Data, jonka Historian REST-vastauksissaan lähettää on aina JSON-muotoista, jonka purkamiseen Power BI soveltuu mainiosti. Ensin ohjelmistolla noudettiin Azuren Blob Storagesta data, jonka Data Factory oli sinne tallentanut. Tämän jälkeen data muokattiin (kuva 16) Power BI:n työkaluilla muotoon, jolla datan visualisointi oli helpompaa.



Kuva 16. Power BI -ohjelmiston kyselyn muokkaus työkalu. JSON-muotoinen data on muunnettu luettavampaan taulukkomuotoon.

Kun data oli saatettu sopivaan muotoon, pystyttiin se visualisoimaan Power BI:n visualisointityökaluilla. Kuvassa 17 on kaksi eri kuvaajaa simulaattoridatalle, toinen sinikäyrälle ja toinen satunnaisarvolle. Lisäksi data esitettiin taulukossa, jonka dynaamisten toimintojen avulla pystyttiin valitsemaan halutut pisteet, jotka kuvaajissa näytettiin.



Kuva 17. Power BI -ohjelmiston visualisointityökaluilla muodostetut kaaviot simulaattoridatasta.

Yksi selkeä käyttökohde on teollisen prosessin eräraportointi, jossa Data Factory on ajastettu juoksemaan erän arvioidun suoritusajan perusteella tai vaikka päivittäin. Power BI:llä luodaan kyselyt, joilla data pilkotaan sopivaan muotoon erittäin ja visualisoinnit tehdään teollisen prosessin arvojen mukaan. Näin rakennetaan raporttipohjia, joille haetaan data Azuren tietovarastoista ja raportit muodostuvat automaattisesti. [9.]

7 Yhteenveto

Insinöörityön tavoitteena oli toteuttaa tiedonsiirto kentältä pilveen Historianin etäkeräilijäohjelmistolla sekä Azure-pilvipalvelun sisällä käyttäen Historianin REST-rajapintaa ja vertailemaan tiedonsiirtomallin toimintaa yksinkertaisempaan, KEPServerEX – IoT Hub -malliin. Päätavoitteena oli saada tiedonsiirto toimimaan testausympäristössä tietoturvallisesti. Toissijaisina tavoitteina oli tutustua Azure-pilvialustaan ja sen tarjoamiin palveluihin ja työkaluihin, joilla tiedonsiirto saataisiin tehtyä sekä Power BI -ohjelmistoon, jolla dataa visualisoitiin. Asetettuihin tavoitteisiin päästiin, joskaan tiedonsiirron tietoturvasta ei tullut täysin aukoton. OAuth2-token tallennettiin salaamattomana Blob Storageen ja tämä vaihe tiedonsiirrossa kaipaa jatkokehitystä.

REST-rajapinnasta ja viestien autentikoinnista sekä Azuren Data Factorystä ja näiden konfiguroinnista luotiin Novotekin sisäiseen käyttöön ohje. Insinöörityön tuloksien avulla Novotek pystyy paremmin myymään Historian-ohjelmistoa IoT-ratkaisuissaan ja tuotekehitystä IoT-ratkaisun ympärillä voidaan jatkaa. Data Factoryn avulla yritys voi myös tarjota uutta palvelua, jossa asiakkaan vanha tietokanta päivitetään uuteen versioon ja vanha data siirretään pilveen analyysia varten tai koneoppimisen tueksi.

Työ oli mielenkiintoinen ja haastava lähtökohtiin nähden. Ennen insinöörityön aloittamista käsitys pilvipalveluista oli rajallinen ja niiden käyttökokemus lähes olematon. Työn anti oli kokonaisvaltainen kuva teollisen internetin tiedonsiirrosta kentältä pilveen.

Microsoftin dokumentaation mukaan Data Factory -työkalu tukee GE Historiania ODBC-rajapinnan avulla. Tarvittavaa ODBC-ajuria ei kuitenkaan GE:ltä kysyttäessä ole edes olemassa, ainakaan avoimesti. ODBC-yhteys pudottaisi REST-rajapinnan Data Factory -työkalun käytöstä pois ja suorituskyky voisi olla RESTiä parempi. Historianin kanssa on myös mahdollista käyttää SQL-rajapintaa Novotekin omalla Novotek Report Plus -tuotteella, jonka sisäänrakennetuilla proseduureilla voidaan automaattisesti noutaa eräraportteihin tarvittava data vain eränumeron perusteella. Data Factory tukee sekä SQL-tietokantoja että näiden valmiiden proseduurien käyttöä.

REST-rajapintaa käytettäessä OAuth2-tokenin tallentaminen Blob Storageen ei ole paras mahdollinen tapa tietoturvan kannalta. Tokenin tallentaminen Azure Key Vault -palveluun ja hallinta Azure Automation -palvelulla on tietoturvallisempi, mutta myös monimutkaisempi ja jatkokehitystä vaativa ongelma.

Lähteet

1 Morgan, Jacob. 2014. A Simple Explanation Of 'The Internet Of Things'. Verkkodokumentti. Forbes. <https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#50d53d941d09>. 13.5.2014. Luettu 16.2.2018.

2 Ashton, Kevin. 2009. That 'Internet of Things' Thing. Verkkodokumentti. RFID Journal. <http://www.rfidjournal.com/articles/view?4986>. 22.6.2009. Luettu 16.2.2018.

3 Collin, Jari & Saarelainen, Ari. 2016. Teollinen internet. Helsinki: Talentum.

4 Evans, Dave. 2011. The Internet of Things How the Next Evolution of the Internet Is Changing Everything. Verkkodokumentti. Cisco IBSG. <https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf>. 4.2011. Luettu 16.2.2018.

5 IPv6 for IoT. Verkkodokumentti. IoT6. <https://iot6.eu/ipv6_for_iot>. Luettu 16.2.2018.

6 Nordrum, Amy. 2016. Popular Internet of Things Forecast of 50 Billion Devices by 2020 is Outdated. Verkkodokumentti. IEEE. <https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated>. 18.8.2016. Luettu 16.2.2018.

7 Towards a definition of the Internet of Things (IoT). 2015. Verkkodokumentti. IEEE. <https://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf>. 27.5.2015. Luettu 16.2.2018.

8 Everything You Need to Know About the Industrial Internet of Things. Verkkodokumentti. GE Digital. <https://www.ge.com/digital/blog/everything-you-need-know-about-industrial-internet-things>. Luettu 17.2.2018.

9 Pirinen, Jukka. 2018. IoT Solutions Manager, Novotek Oy, Vantaa. Keskustelut 19.3.2018−30.3.2018.

10 Gifford, Charlie. 2011. When Worlds Collide in Manufacturing Operations: ISA-95 Best Practices Book 2.0. International Society of Automation.

11 IOT PLATFORMS The central backbone for the Internet of Things. 2015. Verkkodokumentti. IoT Analytics. <http://iot-analytics.com/wp/wp-content/uploads/2016/01/White-paper-IoT-platforms-The-central-backbone-for-the-Internet-of-Things-Nov-2015-vfi5.pdf>. 11.2015. Luettu 19.3.2018.

12 Shodan-hakukoneen Modbus-hakutulos. Verkkodokumentti. <https://www.shodan.io/search?query=port%3A502>. Haku suoritettu 19.3.2018.

13 Rinaldi, John. 2016. Modbus Security. Verkkodokumentti. Real Time Automation. <https://www.rtaautomation.com/blog/modbus-security/>. 4.5.2016. Luettu 19.3.2018.

14 ENISA Threat Landscape Report 2017 15 Top Cyber-Threats and Trends. 2018. Verkkodokumentti. ENISA. <https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-2017>. 1.2018. Luettu 19.3.2018.

15 Woolf, Nicky. 2016. DDoS attack that disrupted internet was largest of its kind in history, experts say. Verkkodokumentti. The Guardian. <https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet>. 26.8.2016. Luettu 19.3.2018.

16 Gertsch, Mia. 2016. Verkkohyökkäys katkaisi talojen lämmityksen Lappeenrannassa ja sulatti jäät Rauman jäähallissa. Verkkodokumentti. Yle. <https://yle.fi/uutiset/3-9278183>. 8.11.2016. Luettu 19.3.2018.

17 Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies. 2016. Verkkodokumentti. US Department of Homeland Security. <https://ics-cert.us-cert.gov/sites/default/files/recommended_practices/NCCIC_ICS-CERT_Defense_in_Depth_2016_S508C.pdf>. 9.2016. Luettu 22.3.2018.

18 What is OPC? Verkkodokumentti. OPC Foundation. <https://opcfoundation.org/about/what-is-opc/>. Luettu 22.2.2018.

19 Unified Architecture. Verkkodokumentti. OPC Foundation. <https://opcfoundation.org/about/opc-technologies/opc-ua/>. Luettu 22.2.2018.

20 Ojala, Rami. 2017. MQTT IoT-protokolla Toiminta ja toteutus. Insinöörityö. Jyväskylän ammattikorkeakoulu. Theseus-tietokanta.

21 Gilchrist, Alasdair. 2016. Industry 4.0: The Industrial Internet of Things. E-kirja. Apress.

22 MQTT Version 3.1.1 OASIS Standard. 2014. Verkkodokumentti. OASIS. <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>. 29.10.2014. Luettu 23.2.2018.

23 AMQP is the Internet Protocol for Business Messaging. Verkkodokumentti. OASIS. <http://www.amqp.org/about/what>. Luettu 22.2.2018.

24 Untangled musings of Roy T. Fielding, About Untangled. Verkkodokumentti. <http://roy.gbiv.com/untangled/about>. Luettu 20.3.2018.

25 RFC2616. Hypertext Transfer Protocol – HTTP/1.1. 1999. The Internet Society.

26 ThingWorx REST API. Updating, Deleting, and Executing Through the API. Verkkodokumentti. PTC. <http://support.ptc.com/cs/help/thingworx_hc/thingworx_6.0_hc/index.jspx?id=thingworx10&action=show>. Luettu 20.3.2018.

27 Historian REST API Reference Version 7.0 SP5. 2017. Verkkodokumentti. GE Digital. <https://digitalsupport.ge.com/servlet/fileField?id=0BE1A000000L4Yi>. 9.2017. Luettu 13.3.2018.

28 George, Sam. 2016. Microsoft introduces new open-source cross-platform OPC UA support of the industrial Internet of Things. Verkkodokumentti. Microsoft. <https://blogs.microsoft.com/iot/2016/06/23/microsoft-introduces-new-open-source-cross-platform-opc-ua-support-for-the-industrial-internet-of-things/>. 23.6.2016. Luettu 22.2.2018.

29 OPC Foundation Announces support of Publish / Subscribe for OPC UA. Verkkodokumentti. OPC Foundation. <https://opcfoundation.org/news/opc-foundation-news/opc-foundation-announces-support-of-publish-subscribe-for-opc-ua/>. 6.4.2016. Luettu 28.4.2018.

30 IS CSV a good alternative to XML and JSON? 2014. Verkkodokumentti. Stack Exchange. <https://softwareengineering.stackexchange.com/questions/224929/is-csv-a-good-alternative-to-xml-and-json>. 22.1.2014. Luettu 21.3.2018.

31 WebSocket Protocol, AlwaysOn™, and Configuring the Edge MicroServer (EMS) in ThingWorx. 2015. Verkkodokumentti. PTC. <https://www.ptc.com/en/support/article?n=CS225419>. 18.1.2018. Luettu 21.3.2018.

32 Kepware kepserverex manual. 2017. Verkkodokumentti. PTC. <https://www.kepware.com/en-us/products/kepserverex/documents/kepserverex-manual/>. Luettu 22.1.2018.

33 Crook, Stacy; MacGillivray, Carrie & Turner, Vernon. 2017. IDC MarketScape: Worldwide IoT Platforms (Software Vendors) 2017 Vendor Assessment. Verkkodokumentti. International Data Corporation. <https://www.idc.com/getdoc.jsp?containerId=US42033517>. 7.2017. Luettu 21.3.2018.

34 ThingWorx Industrial IoT Platform. Verkkodokumentti. PTC. <https://www.ptc.com/en/products/iot/thingworx-platform>. Luettu 21.3.2018.

35 PTC and Elisa Continue Internet of Things Collaboration with the Elisa IoT Innovation Challenge. 2016. Verkkodokumentti. PTC. <https://www.ptc.com/de/news/2016/ptc-and-elisa-continue-iot-collaboration>. 8.9.2016. Luettu 21.3.2018.

36 Introduction to ThingWorx 8. Verkkodokumentti. ptc university. <https://precision-lms.ptc.com/>. Luettu 21.3.2018.

37 Azure Architecture Center. Verkkodokumentti. Microsoft Azure. <https://docs.microsoft.com/en-us/azure/architecture/>. Luettu 23.3.2018.

38 IoT Hub Documentation. Verkkodokumentti. Microsoft Azure. <https://docs.microsoft.com/en-us/azure/iot-hub/>. Luettu 23.3.2018.

39 Device Explorer Documentation. 2018. Verkkodokumentti. Microsoft Azure. <https://github.com/Azure/azure-iot-sdk-csharp/tree/master/tools/DeviceExplorer>. Luettu 23.3.2018.

40 IoT Gateway Manual. 2017. Verkkodokumentti. PTC Inc. Verkkodokumentti. <https://www.kepware.com/en-us/products/kepserverex/advanced-plug-ins/iot-gateway/documents/iot-gateway-manual.pdf>. Luettu 23.3.2018.

41 Stream Analytics Documentation. Verkkodokumentti. Microsoft Azure. <https://docs.microsoft.com/en-us/azure/stream-analytics/>. Luettu 23.3.2018.

42 Azure Storage Documentation. Verkkodokumentti. Microsoft Azure. <https://docs.microsoft.com/en-us/azure/storage/>. Luettu 23.3.2018.

43 Azure Machine Learning Documentation. Verkkodokumentti. Microsoft Azure. <https://docs.microsoft.com/en-us/azure/machine-learning/>. Luettu 23.3.2018.

44 Historian Data Collectors. Verkkodokumentti. General Electric. <http://help.geautomation.com/Historian55/Subsystems/iHistCollMaster/content/ihdc_cover.htm>. Luettu 30.3.2018.

45 Historian Getting Started Guide Version 7.0 SP5. 2017. Verkkodokumentti. GE Digital. <https://digitalsupport.ge.com/servlet/fileField?id=0BE1A000000L4Yg>. 9.2017. Luettu 30.3.2018.

46 RFC 6749The Oauth 2.0 Authorization Framework. 2012. Internet Engineering Task Force (IETF).

47 Reinke, Johann. 2016. Understanding OAuth2. Verkkodokumentti. <http://www.bubblecode.net/en/2016/01/22/understanding-oauth2/>. 22.1.2016. Luettu 6.4.2018.

48 Mills, Chris David. 2018. The general HTTP authentication framework. Verkkodokumentti. Mozilla. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication>. 26.1.2018. Luettu 3.4.2018.

49 Azure Data Factory Documentation. Verkkodokumentti. Microsoft Azure. <https://docs.microsoft.com/en-us/azure/data-factory/>. Luettu 30.3.2018.